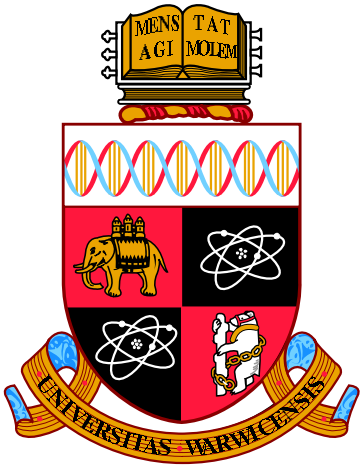


Modified Intelligent Water Drops with Perturbation Operators for Atomic Cluster Optimization

by

RITCHIE MAE TONZO GAMOT

Thesis

Submitted to the University of Warwick

for the degree of

Doctor of Philosophy

Centre for Scientific Computing

April 2016

THE UNIVERSITY OF WARWICK

Contents

List of Tables

List of Figures

Acknowledgments

First of all I would like to express my sincerest gratitude to my supervisor Prof. Mark Rodger for his invaluable guidance throughout the period of this study. Prof. Rodger's expertise and extensive knowledge have always provided me with enlightenment on my confusions and supplied me with ideas that have paved the best direction possible to my study. Prof. Rodger has also been instrumental in my being a recipient of the Chancellor's International Scholarship (formerly *Warwick Postgraduate Research Scholarship*) and the Centre for Scientific Computing (CSC) departmental bursary for which I am truly grateful without which I would not be able to enjoy an international postgraduate research experience as Warwick University provides. To my reviewers, Prof. Roy Johnston of the University of Birmingham and Prof. Ale Troisi of the University of Warwick, thank you for creating a relaxed atmosphere during the viva and I truly appreciate that you enjoyed our conversation. Your invaluable inputs made my thesis a better one.

I am also thankful to the Warwick University as a whole for providing me with the best international research experience I could possibly get. I would like to mention the following offices which I have visited and/or contacted in all of my years here in Warwick. The people in these offices have definitely made my stay in the UK both pleasant and enjoyable: Centre for Scientific Computing, Warwick Accommodation, Warwick Library, Student Funding, Student Finance, International Office, Warwick Arts Centre, IT Services, Student Union, and the Graduate School.

I am also taking this opportunity to extend my warmest gratitude to the members of the FATNode group whom I have shared a myriad of dishes from the various restaurants all over Coventry, Kenilworth and Leamington Spa. I have

both gained and lost 'pounds' (if you know what I mean) during our dinners. This experience has largely increased my cultural experience cuisine-wise.

My social experience would not have been complete without the awesome Filipino group I have met in Warwick. It has always been a relief to talk in Filipino/Visayan from time to time. Our meetings over meals, ciders/beers and coffee have always been the most fun times of my stay in the UK. To all the awesome people from the different walks of life that I have met in the various accommodations I have stayed in and seminars/workshops that I have attended, you all made me feel that I travelled the world by just meeting you. I also got through this PhD journey with my bestfriend Doods who has become my PhD buddy. I could not thank you enough for listening to my endless, senseless badgering day and night.

I cannot express how thankful I am to my mother Magdalina "Dolly" Gamot who has always supported and encouraged me even from 7,000 miles away. Our countless chats, text messages and photo exchanges have kept me closer to home. I am also thankful to my dear brother Mike for providing me support specially when I was preparing to leave for the UK. I also dedicate this thesis to my dear dad Oscar who has passed two years before I started my PhD but I know he would have been proud to know that I have finished another milestone in my life. I am also thanking the non-human member of our family, my beautiful cat Bori, who unfortunately passed while I was on my last few months of writing my thesis. Seeing you in countless photos gave me the breather that I always needed.

Finally, I would like to profusely thank the University of the Philippines (UP) for allowing me to take time off from teaching to pursue a PhD degree in an international reputable university in the UK. Without the UP Foregin Fellowship award and UP Doctral Studies Fund, this study would not have been materialized. The financial support and professional advice were invaluable towards my journey in building a career in the academe. I will always be thankful for the opportunity.

Utmost thanks to the Almighty God above for blessing me with a sound mind to undertake this feat.

Abstract

A modified version of the Intelligent Water Drops algorithm (MIWD) was developed then used to determine the most stable configurations of Lennard-Jones (LJ), Binary Lennard-Jones (BLJ) and Morse Clusters. The algorithm is unbiased in that it uses no *a priori* cluster geometry information or cluster seeds. Results for LJ clusters show that the algorithm is effective and efficient in rediscovering all clusters up to size $N = 104$ with better success rates specially on difficult clusters compared to previous best methodologies reported in literature. Results on more difficult systems, such as the Binary Lennard Jones clusters up to size 50 (with 5 different atomic size ratios) and Morse clusters up to size 60 (with 2 interparticle range potentials), also showed the ability of MIWD to handle more complex systems.

MIWD was then applied to predict the most stable structures of Janus clusters up to size 50 and on size 100 using a LJ potential model with a modulated angular term suited for two-patched Janus particles. Results show that MIWD is able to find well-structured geometries of Janus clusters. It is believed that this has been the first time that a nature-inspired stochastic algorithm and a variant of the IWD algorithm has been applied to the configurational optimization of Janus clusters.

Chapter 1

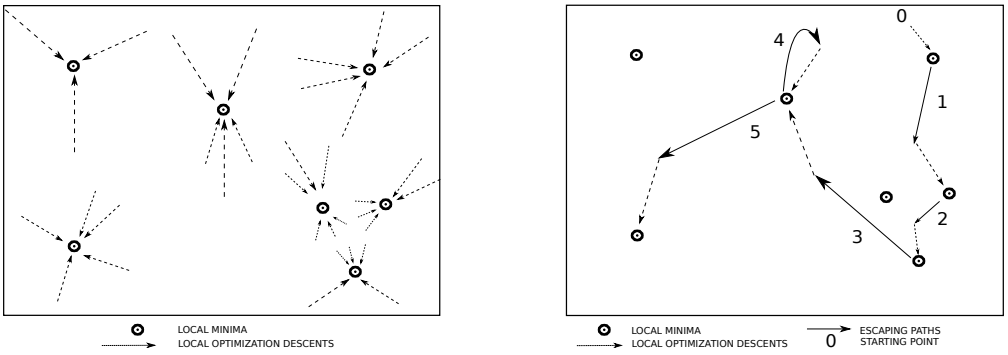
Introduction

This chapter provides a general introduction to global optimization, nature-inspired algorithms, and challenges on atomic cluster optimization including the motivation for this study.

1.1 Global Optimization

The global optimization (GO) problem, as described by Kan *et al* [Kan & Timmer, 1984], is to find the global optimum x_* of a real-valued function $f : R^N \rightarrow R$, such that $f(x_*) \leq f(x) \quad \forall \quad x \in R^N$. To provide a computational boundary, a specified bounded set $S \subset R^N$ containing the global minimum as an interior point is made in advance.

The need for GO before the introduction of electronic computers was not a major concern due to the enormous amount of computational effort it requires. This changed during the advent of computers, where deterministic techniques based on the priniciple of divide-and-conquer, such as the Branch-and-Bound (BB) techniques, were used and first applied to the discrete optimization problems such as the Travelling Salesman Problem (TSP) [Liberti, 2006, and references therein]. BB techniques provide a theoretical guarantee of locating the local minimum but rely heavily on intensive computation to explore the solution space. However, a large portion of real-world problems involve continuous variables with high-dimensionality rendering BB, or deterministic techniques for that matter, even more tedious or requiring problem reformulation. In fact, it was not until 1969 where a deterministic approach to continuous global optimization was introduced by Falk *et al* [Falk & Soland, 1969]. Deterministic optimization methods in the past have mostly been developed to deal with specific cases with only a few dealing with generic nonconvex optimization problems. A brief overview of global optimization by Liberti [Liberti, 2006] chronologically shows that towards the beginning of 1990s, an alternative technique has to emerge to battle



(a) (b)

Figure 1.1: (a) Sampling : Arrows indicate descent to target local minimum (black circle). (b) Escaping : Arrows in solid lines indicate attempt to move to a different basin. Both figures adapted with permission from [Liberti, 2006].

the shortcoming of deterministic approaches.

Stochastic (or probabilistic) techniques, on the other hand, offer an equally capable alternative way of finding solutions to GO problems. A simple property of a deterministic technique given by Weise [Weise, 2009] is that it does not contain instructions using random numbers to decide the next step or how to choose the data. Stochastic techniques proceed otherwise. They employ an element of random choice. Due to this criterion, one cannot always guarantee success; however it has the advantage of practicality in terms computation times. There is also no need for problem reformulation. Essentially, stochastic methods act as a "black box" technique requiring little or no *a priori* information of the problem itself. Due to this inherent characteristic of stochastic methods, a good mix of *sampling escaping* is of paramount importance in the method design [Liberti, 2006]. Sampling (Figure 1.1a) is a way of intelligently distributing more than one agent of solution in search of local optima to different parts of the solution space. Escaping (Figure 1.1b) is implemented after a local optimization procedure has terminated with a local optimum. Escaping, in other words, allows for the exploration of another neighbourhood, ideally a basin that is different from the previously located optimum.

1.1.1 Deterministic versus Stochastic Methods

There have not been many studies comparing deterministic and stochastic methods in the past. However in 1989, certain stochastic algorithms were reported to perform relatively poorly compared with a more deterministic approach. A comparison between deterministic and stochastic algorithms was performed by Blake [Blake, 1989] to solve a nonconvex optimization formulation of a piecewise reconstruction of real-value data. Three variants of simulated annealing (SA) and

a deterministic procedure called the graduated nonconvexity (GNC) algorithm were tested on the "weak string" reconstruction problem [Blake & Zisserman, 1987]. Study results suggested that the simulated annealing approach, especially the Metropolis Heatbath, considered to be the best performing of the three SA variants, required at least 10 times more iterations than GNC. On increasing the levels of noise and scale, SA failed to find a solution while GNC produced a correct reconstruction.

The status of stochastic methods improved in 2005 with a comparison between a deterministic spatial branch-and-bound (sBB) algorithm and a quasi Monte Carlo (QMC) variant of a stochastic multi level single linkage (MLSL) algorithm. Liberti *et al* [Liberti & Kucherenko, 2005] tested these approaches on benchmark test problems systematically chosen to represent different levels of difficulty. They concluded that although the sBB method exhibits superior performance in some instances, the QMC variant of MLSL is capable of locating the global minium with good probability within a reasonable amount of time and is generally more efficient.

A more recent comparison in 2013 of a deterministic and stochastic global optimization method was done on the problem of planar covering with ellipses [Andretta & Birgin, 2013]. In the instances where ellipses were not allowed to rotate, both methods were able to solve the test cases. For the test cases where ellipses were allowed to rotate, the problem became non-convex rendering finding a global solution more expensive. Both methods however were still tested on the latter case with the deterministic approach only able to find optimal solution up to moderately-sized instances of the problem while the stochastic approach also delivered "reasonable" solutions to larger ones.

These successes, although providing limited evidence, allows for motivation to use stochastic techniques in solving GO problems and in this study as well.

1.2 Metaheuristics and Nature-Inspired Algorithms

A family of stochastic techniques, called *metaheuristics*, has been an active area of research. To put everything in perspective, we provide a simple definition of a *heuristic* method first. Heuristics in GO are functions that aid in deciding which of the possible solutions is to be examined next. Furthermore, the set of possible solutions used by heuristics are gathered by the algorithm itself in previous steps. On a different level which can be considered as the next higher step to heuristics is metaheuristics. Although no commonly accepted definition is available, metaheuristics can be thought of as a kind of approximate algorithm that combines

basic heuristic methods in higher level frameworks with the goal of efficiently and effectively exploring a search space. Aside from presenting various definitions on metaheuristics from various authors, Blum *et al* also enumerated several properties of meteheuristics in a survey [Blum & Roli, 2003]. Important properties of metaheuristics that are worth mentioning are the following: Metaheuristics (1) are strategies that "guide" the search process; (2) are approximate and usually non-deterministic; (3) are not problem-specific; (4) incorporate mechanisms to avoid getting trapped in a local minimum and eventually find (near-) optimal solutions. These properties strongly motivate the course of this study and focuses it on a certain class of metaheuristics which will be further discussed in the succeeding sections.

One class of algorithms categorized as meteheuristics is called *nature-inspired algorithms*. Nature here refers to a part of the physical universe which is different from the result of deliberate human design [Steer *et al.*, 2009]. In the following sections, we briefly identify three popular nature-inspired algorithms applied to different types of optimization problems, before describing a fourth, Intelligent Water Drops Algorithm, in Section 1.3 that forms the basis of this thesis.

1.2.1 Genetic Algorithm

Genetic Algorithms (GA), perhaps the most popular of all nature-inspired algorithms, is based on Darwin's Theory of Evolution by Natural Selection. Invented in the 1960s by John Holland [Holland, 1975] and further developed with his students and colleagues in the 1970s, GAs have been applied to many important applications since then. It has been successfully used to predict dynamical systems such as forecasting, management, weather, neuroanalysis, and large-scale modeling [Packard, 1990]. Schulze-Kremer [Schulze-Kremer, 1994] also applied it to predicting protein structure. An amino acid sequence of the Crambin protein was taken and a GA was used to search possible structures that would fit well with Crambin's amino acid sequence. A GA was also used as an aid to another biologically-inspired algorithm called neural networks (NN) by a GA-evolved weights of a fixed NN instead of using back-propagation to the sonar image classification problem [Montana & Davis, 1989]. It was also applied to evolving NN architectures (as opposed to fixed NN) by automatically determining the optimal design for a specific application thru GA rather than deciding the architecture ahead of time by the programmer thru guesswork [Miller *et al.*, 1993]. More recently Johnston [Johnston, 2003] successfully developed the Birmingham cluster GA program for optimising cluster geometries. Birmingham Cluster GA was able to search low energy isomers for a varieity of clusters such as Morse

clusters, ionic clusters, and nanoalloy clusters (Cu-Au,Ni-Al, Pd-Pt).

1.2.2 Ant Colony Optimization

Another successful bio-inspired algorithm is called the *Ant System* (AS). Proposed by Dorigo *et al* [Colorni *et al.*, 1991;Dorigo *et al.*, 1996], this algorithm is based on behaviours exhibited by colonies of ants when locating and collecting food. Amongst its first applications were on symmetric and asymmetric TSP, quadratic assignment problem and job-shop scheduling. Artificial Bee Colony (ABC) or Ant Colony optimization (ACO), as it is also called, has been tested on computing pixel classification for image segmentation [E. Cuevas & Perez, 2013]. ABC was used to calculate parameters of a Gaussian mixture model that approximates an image's 1-D histogram. The algorithm showed more robust performance regardless of initial conditions and was compared with commonly employed methods in Gaussian mixtures such as the Expectation-Maximization and Levenberg-Marquardt algorithms. A multi-colony Ant System version was also used for goods transportation, and gave numerical results that outperformed traditional approaches significantly [Doerner *et al.*, 2003]. In each colony, which deals with problems with partially conflicting goals, information is gathered heuristically to construct a solution. An information spillover, termed as *ant spies*, between colonies also allows performance to be shared by the superior performing colony with the inferior one. Another variant of AS to dynamic 25- and 100-city TSP (i.e. distances between cities were treated as travel times and jams occur in certain times on a particular connection by assigning a longer travel time) "performed reasonably well" [Eyckelhof & Snoek, 2002]. Global and local *shaking* was introduced in the variant where pheromone levels were smoothened in a way to allow selection of other roads when a traffic jam occured on the highest-pheromone-level road. Furthermore, ACO was also successfully tested on protein folding problem [Shymgelska & Hoos, 2005] and protein-ligand docking [Korb *et al.*, 2007]. In 2D and 3D hydrophobic polar (HP) protein folding using discretized amino acids, ACO "performed fairly well" compared to benchmark heuristics on benchmark instances except PERM (best known algorithm for 2D and 3D HP protein folding problem). On 3D HP protein folding problem, ACO scaled worse than PERM but it was able to find different ensemble of native conformations. ACO also encountered less difficulty in folding sequences whose native states contained structural nuclei located in the middle rather than at the end, as well as sequences with structures in which the ends interact. Korb, *et al* developed a docking algorithm, termed PLANTS or Protein-Ligand ANT System [Korb *et al.*, 2007], that was used to predict the complex structure of a small ligand with a protein which is useful for rational design of new drugs. Following the combinatorial nature of

ACO, the variables involved were discretized according to interval values of the degrees of freedom (translational, rotational and torsional). To aid further in the search, a local search which works on the continuous search space using an algorithm by Nelder and Mead [Nelder & Mead, 1965] was applied. In comparison with the state-of-the-art docking program GOLD, PLANTS showed higher pose prediction accuracy at similar or at lower docking times.

1.2.3 Particle Swarm Optimization

Gaining ground in the area of optimization is an algorithm called Particle Swarm Optimization (PSO). Developed by Kennedy and Eberhart in 1995, it is based on the social behaviour of bird flocking and fish schooling. It was initially tested on adjusting weights to train feedforward multilayer perceptrion neural network and showed that it can train weights as effectively as the usual error backpropagation method [Kennedy & Eberhart, 1995]. In the same study, using the benchmark function Schaffer f6, PSO performed comparatively well to GA. A survey on advances on PSO by Jin *et al* [Jin & Rahmat-Samii, 2007] discussed the application of PSO to aperiodic antenna array design using different PSO versions. Real-number RPSO has been used in non-uniform antenna array design while binary PSO was used to design array thinning. Furthermore, multi-objective PSO (MOPSO) and multi-ojbective Binary PSO (MOBPSO) were also attempted in optimizing another design element such as beamwidth. Two variants of PSO were also applied to distributed odor source localization in a dynamic environment. PSO was used to traverse the plume towards the source. The first variant, Detect and Respond PSO, was able to solve the odor source localization but the restart mechanism when the environment changes caused information obtained in the search to that point to be lost. Thus, a second variant called Charged PSO was developed. In Charged PSO, particles which they called *robots* were given "charge" based on Coulomb's law to act as a repulsion function. The "charging" of robots was done to enable exploration of different regions of the search space thus essentially increasing diversity. The second variant successfully solved the odor source localization with more robust performance than the initial version [Jatmiko *et al.*, 2005]. An attempt to determine the RNA secondary structure was also done using set-based PSO (setPSO) on 4 different RNA sequences [Neethling & Engelbrecht, 2006]. Although the objective function, which was based on free energy minimisation, used in setPSO was not sufficient to base a prediction of the secondary structure, setPSO was able to find optimal or near optimal solutions. PSO was also combined with ACO (PSO/ACO) to classify hierarchical biological data. PSO was enchanced to cope with categorical attributes of data classification using the pheromone-based mechanism of ACO. Compared to an existing

Discrete PSO (DPSO) for multi-valued categorical attributes, PSO/ACO outperformed DPSO on higher EC (enzyme commission numbers) levels (levels with lesser number of classes) while performing comparatively well, albeit with slightly lower accuracy, at deeper EC levels (levels with larger number of classes). Rule discovery for PSO/ACO was shorter compared with DPSO, as well as making it easier to interpret [Holden & Freitas, 2005].

1.2.4 Nature as Optimizers

There are several other nature-inspired algorithms such as the classic backpropagation neural networks (BPNN) which have been successfully applied to several pattern recogniton and classification problems[Ahmad *et al.*, 2011;Hassan *et al.*, 2010;Singh, 2013]. There are more recent algorithms which are gaining success in the field of optimization such as the bat algorithm, shuffled-frog leaping algorithm, firefly algorithm, cat swarm optimization, cuckoo search, invasive weed optimization, flower pollination algorithms, bacterial colony optimization and intelligent water drops algorithm, among others. This list can go on and so readers are directed to a book by Xing and Gao [Xing & Gao, 2013] for a rough guide to over 100 clever algorithms.

It is evident that, even in the few studies mentioned in the preceeding paragraphs, nature-inspired algorithms are capable of solving various real-world optimization problems. Some have been extensively applied to various problems and some are just starting to gain attention. The next section focuses on one such algorithm which has been in the optimization arena for less than 10 years; however numerous researchers are realizing its value and have focused their attention on improving it and using the variants to solve real-world applications.

1.3 Intelligent Water Drops Algorithm

1.3.1 Background

A river can be seen as a collection of water drops flowing from higher elevations of land, such as mountains, to lower altitudes under the influence of gravity. As each water drop moves from one part/point of the stream to the next, it will encounter barriers such as rocks and soil which affects its movements. As each water drop is influenced by gravity, there is an associated velocity to it. The speed at which each water drop moves is affected by the gradient or the roughness of the riverbed terrain. In addition, the water drop's velocity can have direct influence on the riverbed's properties. In short, there is an interplay between the water drops in the river and the riverbed in which the water drops flow. Hamed Shah-Hosseini

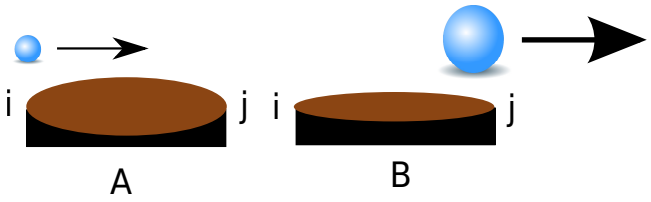


Figure 1.2: An IWD (blue circle) gathers soil (ellipse) as it flows from point i to point j while $path(i, j)$ loses an amount of soil indicated by the thinner ellipse.

developed an algorithm based on this idea and called it *Intelligent Water Drops Algorithm* (IWD) [Shah-Hosseini, 2007].

The following assumptions make up the basic ideas of the IWD algorithm:

1. An IWD begins with an initial velocity, $velocity^{IWD} = initVel$, and an initial soil, $soil^{IWD} = initSoil$, both of which changes in value as they travel in the environment (Figure 1.2).

2. An IWD with higher velocity will gather more soil than a slower IWD (Figure 1.3).

3. The velocity of an IWD is increased by an amount non-linearly proportional to the inverse of the soil content between the path of points i and j ($soil(i, j)$). This allows IWDs passing in paths with lesser amount of soil to gather more speed (Figure 1.4).

4. The amount of soil in a path is decreased as an IWD passes. The amount of soil gathered by the IWD or removed from the path is non-linearly proportional to the inverse of the time required to traverse the path.

5. The length of time an IWD traverses a path follows the simple laws of physics for linear motion. The time interval is proportional to its velocity and inversely proportional to the distance.

6. Paths with lesser amount of soil have higher chances to be selected by an IWD.

1.3.2 Problem Representation

The IWD algorithm gets a representation of the problem in the form of a graph (N, E) with the node set N and edge set E. For TSP the number of nodes is the number of cities N_C and the number of edges is $N_C(N_C - 1)$. The IWD is a multi-agent algorithm where each IWD of the population begins constructing its solution, on different starting nodes, by gradually travelling on the nodes

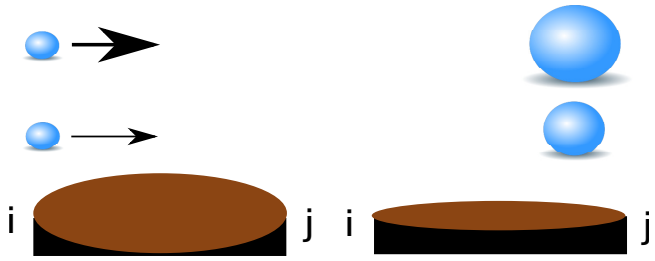


Figure 1.3: Two IWDs following the same path with different velocities (indicated by fatness of arrows) carry different amounts of soil (indicated by area of the circle) at the end of path(i,j). The IWD with the higher velocity carries more soil.

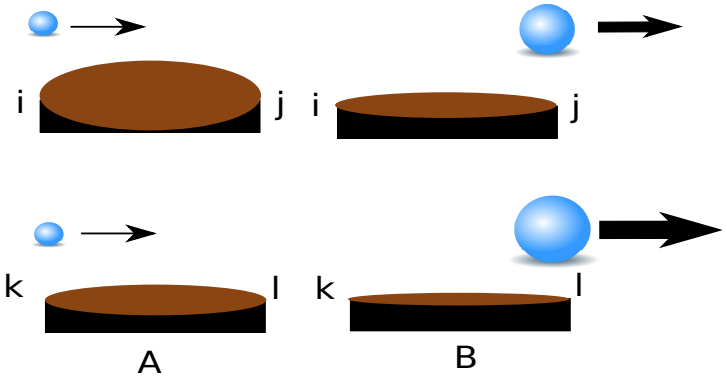


Figure 1.4: An IWD travelling on a path with lesser soil, path(k,l), will gather more soil and generate higher velocity at the end of the path .

of the graph along the edges of the graph until the IWD finally completes its solution. An IWD is thus a representation, at each iteration, of the nodes included into the solution so far. An iteration is considered complete when all IWDs have completed their solutions. A complete iteration in TSP also means each IWD gathering N_C number of nodes. An IWD for TSP can be represented as $IWD_j = \{c_1, c_2, ..., c_i, ..., c_{N_C}\}$ where $c_i \in N$ and $c_i = [x_i, y_i]^T$, N_C is the total number of cities, $3N_C$ the total number of decision variables, $j \in [1, N_{IWD}]$, and N_{IWD} the total number of IWD agents in the population.

1.3.3 Algorithmic Details

Now that the assumptions for IWD have been introduced, the formal algorithm as presented in the first paper in which it appeared [Shah-Hosseini, 2007] and applied to TSP follows below. Since it is based on the TSP, the points i are referred to as cities and the aim is to design the shortest *tour* that visits all cities. Flowchart of the algorithm is also presented in Figure 1.5 whilst a schematic diagram of how IWD progressively gathers components of the solution is shown in Figure 1.6.

1. Initialize the static parameters: set the number of water drops N_{IWD}, the number of cities N_C, and the Cartesian coordinates of each city i such that $c(i) = [x_i, y_i]^T$. The number of cities and their coordinates depend on the problem at hand while the N_{IWD} is set by the user. The velocity updating parameters a_v, b_v, and c_v and soil updating parameters a_s, b_s, and c_s are also initialized in this step. The soil content of the link between every two cities i and j is initially set to $soil(i, j) = InitSoil$. The initial velocity of each IWD is set to $velocity^{IWD} = InitVel$. Both $InitSoil$ and $InitVel$ are set by the user. The best tour T_b is initially unknown and so its length set to infinity at the start ($Len(T_B) = \infty$).

2. Initialize the dynamic parameters: For every IWD, a list of visited cities is created, $V_C(IWD)$. Initially $V_C(IWD) = \{\}$. The initial soil of each IWD, $soil^{IWD}$, is set to zero.

3. For every IWD, randomly select a city and place that IWD on that city.

4. Update $V_C(IWD) = \{\}$ of each IWD to include that city.

5. For each IWD, choose the next city, j, to be visited from the current city, i, using the following probability:

$$p_{i,j}^{IWD} = \frac{f(soil(i,j))}{\sum_{k \notin V_C(IWD)} f(soil(i,k))} \tag{1.1}$$

such that

$$f(soil(i,j)) = \frac{1}{\varepsilon_s + g(soil(i,j))} \tag{1.2}$$

and

$$g\left(soil\left(i,j\right)\right) = \begin{cases} soil\left(i,j\right) & \text{if } \min\limits_{l \notin V_C^{IWd}} soil\left(i,l\right) \geq 0; \\ soil\left(i,j\right) - \min\limits_{l \notin V_C^{IWd}} soil\left(i,l\right) & else. \end{cases} \tag{1.3}$$

Here ε_s is a small positive number chosen to be large enough to prevent division by zero in the function $f(soil(i,j))$. The function min returns the minimum value among all available values for its arguments. The function $g\left(soil\left(i,j\right)\right)$ is a conditional function dependent on the value of the $\min\limits_{l \notin V_C^{IWd}} soil\left(i,l\right)$. Since the value of $soil(i,j)$ belonging to a path traversed by IWD may always be decreased if it is a desirable connection (See step 8), $soil(i,j)$ can get a value below 0.0. The strength of the connectivity between cities i and j is thus not the numerical value of $soil(i,j)$ but its difference from the most negative $soil(i,j)$ value, if it exists.

6. For each IWD moving from city i to city j, update its velocity as follows:

$$velocity^{IWD}(t+1,i,j) = velocity^{IWD}(t) + \frac{a_v}{b_v + c_v soil(i,j)} \tag{1.4}$$

such that $velocity^{IWD}(t+1,i,j)$ is the updated velocity of the IWD whereas $velocity^{IWD}(t,i,j)$ is the velocity of the IWD in the last iteration. An IWD that passes through a path with a larger amount of soil will have smaller velocity while a path with a smaller amount of soil will give the IWD a larger velocity.

Equation 1.4 allows an IWD to generate higher(lower) velocity if the soil content on the path is low(high).

7. For each IWD, compute the amount of soil, $\triangle soil(i,j)$, that it picks up from the path between cities i and j.

$$\triangle\, soil(i,j) = \frac{a_s}{b_s + c_s time(i,j;velocity^{IWD})} \tag{1.5}$$

where

$$time(i,j;velocity^{IWD}) = \frac{HUD}{max(\varepsilon_v, velocity^{IWD})} \tag{1.6}$$

The function $time(i,j;velocity^{IWD})$ computes the time to travel from city i to city j given $velocity^{IWD}$. The dividend HUD represents the *heuristic undesirability factor* or the undesirability of an IWD to move to city j from city i. For TSP, the appropriate HUD_{TSP} is $\|c(i) - c(j)\|$. The function $max(.)$ returns the maximum value among its arguments which is used to threshold negative or zero velocities to a small positive number ε_v.

8. For each IWD, update the soil content of the path traversed by the IWD and the soil that the IWD carries using the following formulae:

$$soil(i,j) = (1-\rho)soil(i,j) - \rho \triangle soil(i,j) \tag{1.7}$$

$$soil^{IWD} = soil^{IWD} + \triangle soil(i,j) \tag{1.8}$$

where the parameter ρ is a small positive number less than 1.0.

9. For each IWD, complete its tour by repeating steps 5 to 8. Once all IWD have completed their tours, calculate the length of tour for each IWD, $Tour^{IWD}$. Find the IWD that has the tour with the minimum length, T_M, in this iteration.

10. Update the soil content of the paths associated with the current minimum tour, T_M, of the IWD using the formula:

$$soil(i,j) = (1-\rho)soil(i,j) + \rho\frac{2soil^{IWD}}{N_c(N_c-1)} \quad \forall(i,j) \in T_M \tag{1.9}$$

11. If the minimum tour T_M is shorter than the best tour T_B found so far, update the best tour by $T_B = T_M$ and $Len(T_B) = Len(T_M)$. This step completes one iteration of the algorithm.

12. Repeat from step 2 until maximum iterations is reached or termination criterion is satisfied. As the algorithm ends, solution T_B is returned with fitness or quality $Len(T_B)$.

In Figure 1.6, we show a schematic diagram of the application of IWD to finding solutions involving four cities. Red and numbered circles represent randomly selected initial cities in step 3 and the IWD identifier, respectively. In

this particular diagram, there are four solutions/agents searching through the solution space in parallel. The search starts from the red circles and, using the probability in Equation 1.1, proceeds to find the remaining components marked by arrows. In the diagram, same colored circles including the red circle to which they are connected to, make up one possible solution to the problem.

1.3.4 Recent Advances on IWD

A few years after the original IWD for discrete optimization was proposed, the author Shah-Hosseini proposed a continuous version which he called IWD for continuous optimization (IWD-CO) [Shah-Hosseini, 2011]. In IWD-CO, IWDs were represented as bit strings and a mutation operator was added as a local search. Tested on six benchmark functions, IWD-CO had satisfactory results. In the paper, the author encouraged future research on different encodings and devising better local search schemes. Later on this was used in a combined economic and emission dispatch (CEED) [Nagalakshmi *et al.*, 2011] which aimed at minimizing fuel and emission cost of generating units subject to some constraints. Emission load dispatch refers to minimizing contaminant emissions from a thermal plant. The Economic load dispatch, on the other hand, tries to minimize the fuel cost by satisfying the load and generator bounded constraints. Compared to GA and PSO, IWD-CO for CEED produced better average convergence time, execution time and number of iterations in finding the optimum solution. In another study, IWD was interspersed with BPNN (as a fitness evaluator) and called Neural-IWD [Hendrawan & Murase, 2011]. It was used to determine water stress in cultured Sunagoke moss by developing a machine vision-based precision irrigation system. This was achieved by using Neural-IWD to find the most significant set of Textural Features suitable for predicting water content of cultured Sunagoke moss. Comparative analysis of the results showed that prediction performance of Neural-IWD was superior to Neural-GA, Neural-DPSO, and Neural-SA. An improved IWD, updating the soil content of the neighbouring paths along the route instead of just the paths along its route, was also used to improve construction of a data aggregation tree (DAT) of wireless sensor networks (WSN) [Hoang *et al.*, 2012]. This improvement was done to increase the probability of selecting optimum aggregation nodes. Compared with ACO, the improved IWD showed better DAT with smaller number of edges. Compared with the basic IWD, improved IWD provided better performance in saving energy for WSNs. Two recent studies modified the selection procedure of IWD and applied on various optimization problems. A study by Islam and Rahman [Islam & Rahman, 2013] applied an ACO-inspired selection procedure and applied it to a real-life waste collection problem (an extension of vehicle routing problem with time windows). Tested

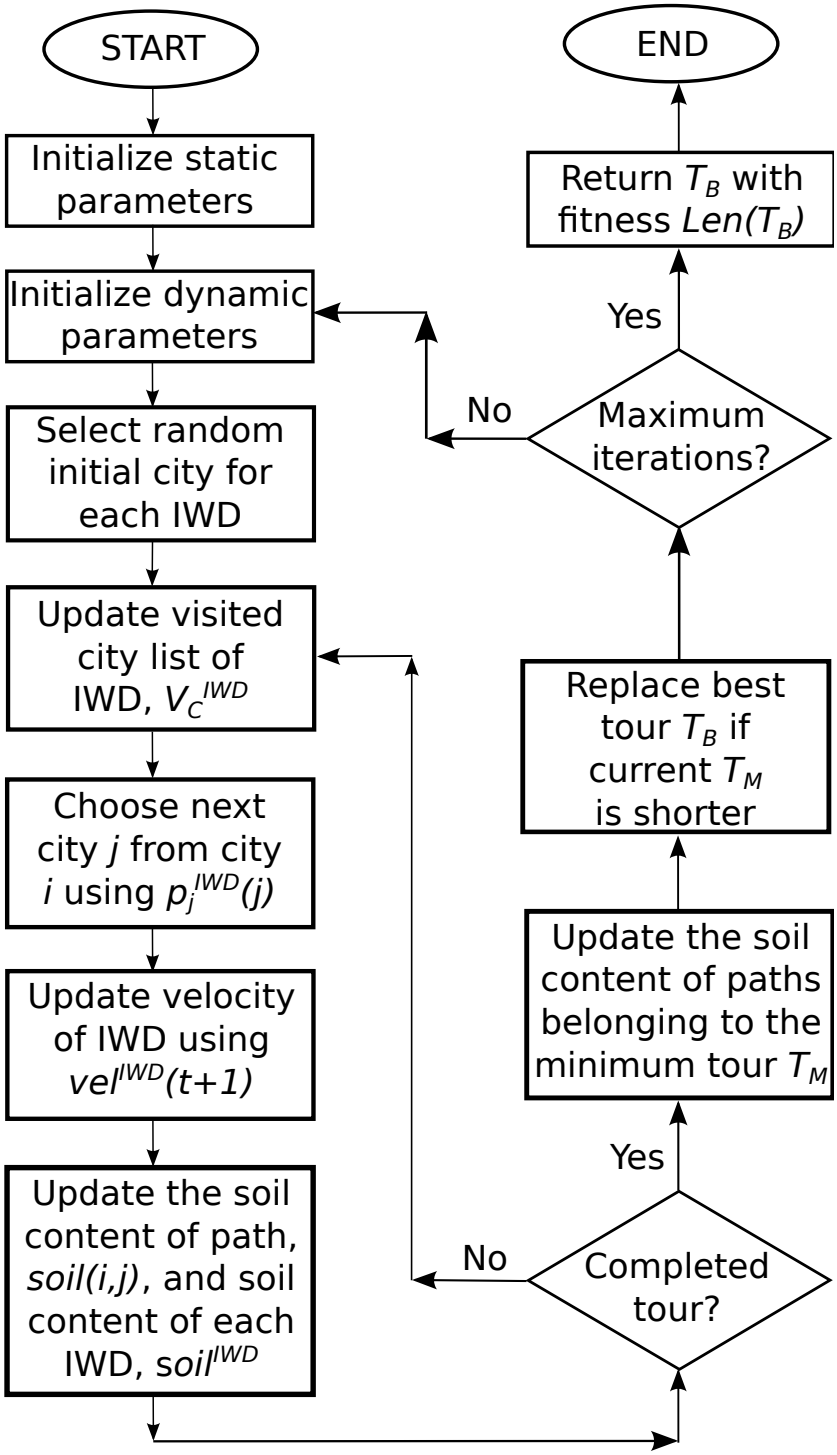


Figure 1.5: Flowchart of the original IWD based on its initial application to TSP.

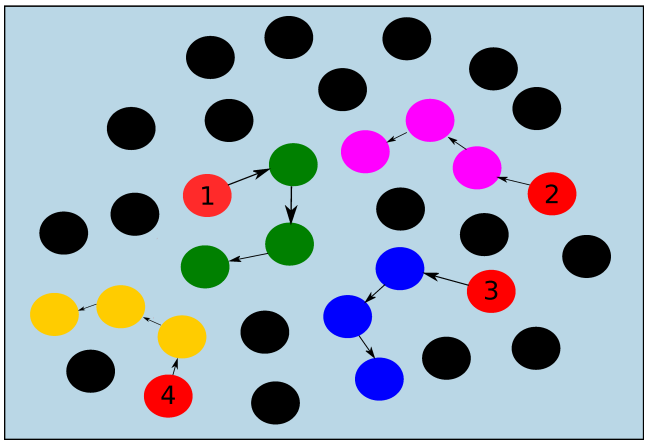


Figure 1.6: Schematic diagram of the progressive building of 4 solutions/IWD agents containing four components or variables using IWD.

on 10 problem instances, the modified IWD outperformed all results from two algorithms in the literature except for one test instance and was better compared to the original implementation of IWD. Alijla *et al* [Alijla *et al.*, 2014] proposed linear-ranking and exponential-ranking based IWD selection methods for the TSP, multiple knapsack problem and rough set feature subset selection. Between the two selection methods, the exponential-ranking based method was able to preserve diversity and ultimately improved algorithm efficiently.

Improvements and modifications to the IWD showed very promising results for various optimization problems. IWD is clearly a capable algorithm not only for discrete problems but also for continuous optimization problems. Performance can also be improved by modifying the selection schemes for solution construction and hybridizing IWD with proven existing operators. The modifications done in the past on IWD motivated the author of this study to follow a similar objective and devise modifications to IWD to allow for its application to a problem that is conceptually simple, but difficult to solve computationally: determing the optimum conformation of a cluster of atoms or molecules.

1.4 Atomic Cluster Optimization

The term *cluster*, sometimes *microcluster*, has been used for various different systems whose common feature is the finiteness of their size. Clusters can be an aggregate of atoms, ions or molecules [Wales, 2003;Hoare, 1979]. Though clusters are usually considered small (two to a few hundred atoms), their surface contains an appreciable portion of the units to be noticed or considered important [Hoare, 1979]. In the succeeding paragraphs and sections, a cluster is used to describe an aggregate of atoms or particles; aggregates of molecules are not considered

explicitly.

The goal of structural optimisation of atomic clusters (or sometimes referred to as the molecular conformation problem), where each atom interacts with all other atoms through different components of the potential energy function, is to *identify the relative atom positions corresponding to the global minimum potential energy for the cluster* [Locatelli & Schoen, 2001;Pullan, 2010].

In chemical physics, interest in efficient global optimisation methods stems from the interest in finding the lowest energy configuration of a molecular system. It is believed that the native structure of a protein, or the most stable structure, is structurally related to the global minimum of its free energy surface [Wales & Doye, 1997;Zhou *et al.*, 2005]. One may assume that the geometry most likely to be discovered experimentally corresponds to the global minimum of the PES in the configuration space of the cluster [Hartke, 1999]. Aside from the geometries corresponding to the global minimum, even low-lying minima are also helpful towards a better understanding of the features of the PES [Hartke, 1999]. If there is a way to locate the global minimum of a cluster, e.g. from the primary amino acid sequence, insights into the nature of protein folding could be easily gathered and allow biochemists to save hours of laboratory work [Wales & Doye, 1997]. Computational biochemists are also interested in the design of molecules for specific applications such as the development of enzymes for removal of toxic wastes, development of new catalysts for material processing and design of anti-cancer agents which all depends on the accurate determination of the structure of biological macro-molecules [Meza & Martinez, 1994].

Atomic cluster optimization has been proven to belong to the class of Non-deterministic Polynomial-time hard (NP-hard) problems requiring a heuristic method of solution to solve them [Wille & Vennik, 1985;Greenwood, 1999]. Any configuration can be allowed to relax to the adjacent minimum in the PES but unless this configuration lies in the proper catchment basin, this configuration will not correspond to the absolute global optimum. It is thus impractical to proceed with an undirected search for all local minima of the potential function in order to locate the global minimum, except for very small clusters [Northby, 1987;Xue, 1994b]. A pure random walk in the configuration space is also unlikely to find the global minimum [Niesse & Mayne, 1996].

The following sections will introduce the different systems used to test the algorithm in this study. The first system is the Lennard Jones (LJ) clusters (Section 1.4.1) which has remained a benchmark for testing effectivity of optimization algorithms such as GA [Deaven *et al.*, 1996], basin-hopping [Wales & Doye, 1997], Conformational Space Annealing [Lee *et al.*, 2003], continous extremal optimization [Zhou *et al.*, 2005], ant colony optimization with monte carlo

sampling method [Tomson & Greenwood, 2005] and particle swarm optimization [Deep *et al.*, 2011]. The LJ clusters had been intensely studied than any other system and so constitute a test set against which new methods are validated. The second system is the Binary LJ clusters (Section 1.4.2) offers a more challenging task for new optimization algorithms due to the additional compositional component of the problem. The third system is the Morse clusters (Section 1.4.3) which provides a way to test optimization algorithms their ability to find different structural behaviors across different cluster sizes. The last system is the Janus clusters (Section 1.4.4) which provides a different challenge to any new optimization algorithm as interaction between Janus particles is not just spatial but also orientational. Furthermore, to date there has been no study using a nature-inspired algorithm to find the global optima of Janus clusters.

1.4.1 Lennard Jones Clusters

The LJ cluster potential energy function is perceived to be a reasonably accurate model of low temperature clusters of heavy rare gas atoms such as Ar or Xe [Leary, 2000]. These are physical systems which are accessible to experimental measurement and so it is possible to compare various respects of the results from global optimization computations with laboratory measurements. Protein folding model energy functions includes the LJ pair potential term which makes the LJ cluster problem, or 12-6 or 6-12 potential energy model, an appropriate test for algorithms potentially applicable to protein folding.

Conformational optimization of an N-atom LJ cluster with minimum potential energy, when all atoms are of the same type, can be expressed as follows:

For $i = 1, \ldots, N$ find points $p_i = (x_i, y_i, z_i)$ denoting the atom coordinates in 3D space, minimizing

$$V_{LJ}(r_{ij}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} 4\varepsilon_{ij}[(\sigma_{ij}/r_{ij})^{12} - (\sigma_{ij}/r_{ij})^{6}] \tag{1.10}$$

where $r_{ij} = \sqrt[2]{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$ denotes the Euclidean distance between points p_i and p_j [Barron *et al.*, 1997]. The parameters ε_{ij} and σ_{ij} are LJ parameters and are different for different types of interacting particles. ε_{ij} is the depth of the potential well, while σ_{ij} is the finite distance at which the inter-particle potential is zero (see Figure 1.7). The LJ potential is strongly repulsive at short distances and passes through 0 at $r_{ij} = \sigma$, i.e. $V(\sigma) = 0$. Its minimum is at $r_{min} = 2^{1/6}\sigma$, i.e. $V(r_{min}) = -\varepsilon$. For a single pair (p_i, p_j) of particles and where $\sigma_{ij} = \varepsilon_{ij} = 1.0$, graph of the LJ pairwise potential energy is shown in Figure 1.7.

The global minima of most LJ clusters are based on incomplete Mackay

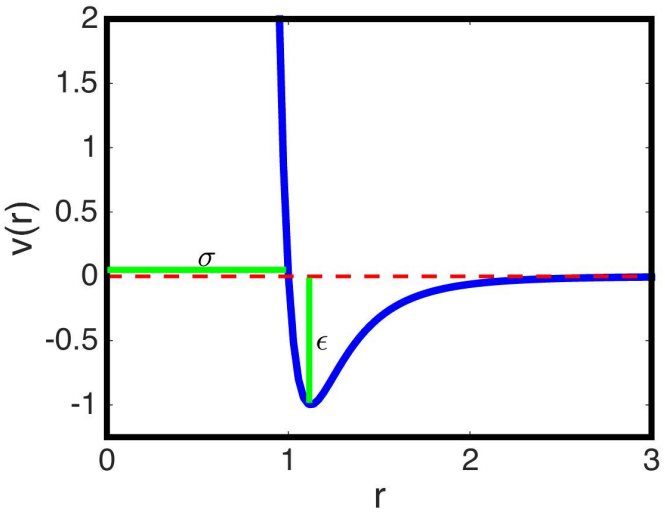


Figure 1.7: A graph of energy, $V(r_{ij})$, versus distance, r, under the 12-6 potential.

icosahedra packing with complete icosahedra occuring at $N = 13, 55, 147, 309, \ldots$. A complete Mackay icosahedron is composed of 20 slightly distorted face-centered-cubic tetrahedra sharing a common point. This type of icosahedron is favoured over other structures due to the larger number of nearest neighbour contacts. The exceptions to this are the nonicosahedral global minima of LJ clusters $N = 38, 75-77, 98, 102-104$. The geometries of these nonicosahedral LJ clusters are truncated octahedron for $N = 38$, Marks Decahedra for $N = 75-77, 102-104$ and Leary Tetrahedron for $N = 98$ [Wales, 2003].

Finding the structure with the minimum LJ energy is a complicated problem due to the presence of hundreds or thousands of local minima in the potential function [Zhou *et al.*, 2005]. In fact, multiplicities in isomers (excluding enantiomorphs or mirror images) for LJ minima from N = 6 to 13 grow as follows : $2, 4, 8, 18, 57, 145, 366, 988$. An exponential function fitted to these pairs of values was generated to be $g(N) = exp(-2.15 + 0.36N + 0.029N^2)$. Extrapolated values for $N = 14$ and 15 show that $g(14) = 3279$ and $g(15) = 10,753$. This is an indication of a very steep rise in isomer counts with size [Hoare, 1979].

1.4.2 Binary Lennard Jones Clusters

The Binary LJ pairwise interaction follows the same potential model as its homonuclear counterpart (Equation 1.10) but this time atoms i and j take on one of two atom types, denoted A or B. The properties of the mixture are specified by 6 parameters, ε_{AA}, ε_{BB}, ε_{AB}, σ_{AA}, σ_{BB}, and σ_{AB}, although it is common to adopt a mixing rule for the cross terms: $\varepsilon_{AB} = \varepsilon_{AA} = \varepsilon_{BB}$, reducing this to 4 parameters. Further, the system follows a scaling behaviour that amounts to

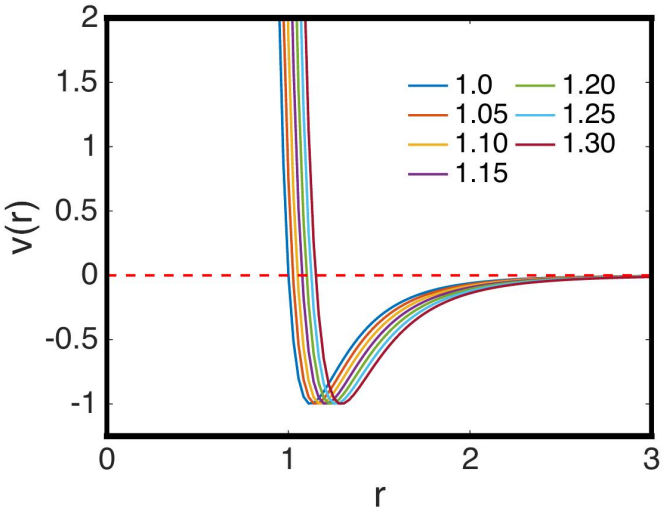


Figure 1.8: A graph of energy, $V(r_{ij})$, versus distance, r, for two atoms with different types under the Binary 12-6 potential for varying atomic size ratio, σ_{BB}.

choosing one (σ, ε) pair to define the length and energy units; in this case we choose particle A for this, so that $\varepsilon_{AA} = \sigma_{AA} = 1.0$. Thus the nature of the mixture is defined by just 2 potential parameters, ε_{BB} and σ_{BB}, along with state variable such as composition. This leaves with σ_{BB} to be chosen, a parameter considered to be related to the van der Waals radius of each atom. The values for this remaining parameter are chosen to be usually between the range 1.0 and 1.3, inclusive. Plots of energy of two atoms with different types using the rest of the coupling coefficients shown above are presented in Figure 1.8. σ_{BB} at 1.0 shows a graph equivalent to the homonuclear LJ graph while the remaining six values (1.05 to 1.30) shows longer inter-atomic distances at the deepest potential well as atomic ratio increases.

An analysis of the changes in Binary LJ stable structures with different atom sizes for σ_{BB} varying from the range 1.0 to 1.3 showed that for the majority of this parameter space, the global minima are polytetrahedral [Doye & Meyer, 2005]. For the homogenous LJ clusters, polytetrahedral structures of size 30 and above are disfavoured due to a greater strain of energy. Nonpolytetrahedral structures in Binary LJ clusters only appear in large clusters with smaller size ratio (i.e. type A and B atoms are almost similar in size).

Binary cluster optimization is a more challenging task for the theoretician due to a larger number of minima than its homonuclear counterpart owing to the fact that "homotops" exist [Doye & Meyer, 2005, 2006], i.e. isomers with the same geometric structure but with different distributions of the two types of atoms, and that the cluster composition is an additional criteria to consider

[Doye & Meyer, 2006]. In fact, clusters up to size 100 atoms would require 5050 different global minima to be discovered [Doye & Meyer, 2005, 2006]. A short but simple proof of the complexity of heteronuclear clusters such as the Binary LJ clusters shows that it is equivalent to solving an instance of the Travelling Salesman Extension which is itself an NP-Hard problem [Greenwood, 1999].

1.4.3 Morse Clusters

A different potential from the LJ potential model, called the Morse potential, was also used as a test system for MIWD. Morse potential may be used to describe either long-range interactions such as alkali metal clusters or short-range potentials such as arising between C_{60} molecules. The Morse potential was also shown to apply to the description of the properties of cubic metals [Girifalco & Weizer, 1959]. Morse clusters exhibit different morphologies namely, icosahedra, decahedra and close-packed clusters. These morphologies are also exhibited by the LJ clusters. However, for LJ clusters with up to $N < 1600$, icosahedra are most stable; from $N = 1600$ up to $\sim 10^5$, decahedra are most stable and beyond $N = 10^5$, face-centered cubic (fcc) clusters [Doye & Wales, 1997]. For LJ cluster size < 110, there is only one global minimum that is an fcc truncated octahedron and at least six optimal structures based on Marks decahedra. The LJ potential clearly exhibits less diversity of structural behaviours in the small-size regime. Morse clusters, on the other hand, transition from icosahedral to decahedral to close-packed clusters "abruptly" within the small-size regime. This characteristic provides a global optimisation algorithm an alternative test sytem to verify its effectivity on a more rugged landscape.

The energy of a Morse cluster is represented by the N-particle pair-wise additive potential [Pereira *et al.*, 2008].

$$V_{Morse}(r_{ij}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} D_e[e^{\alpha(1-r_{ij}/r_e)}(e^{\alpha(1-r_{ij}/r_e)} - 2)] \tag{1.11}$$

where r_{ij} is the Euclidean distance between coordinates of atoms i and j, r_e is the equilibrium bond length, D_e is the bond dissociation energy, and α is the range of the potential. Similarly to the LJ potential, the Morse potentially contains a parameter that defines the depth of the pair potential well (D_e) and the location of the minimum (r_e). It does, however, also contain a third parameter that determines the range of the interaction (α) and so presents a more complex parameter space in which to characterise cluster geometries.

Different structural behaviours are exhibited as a function of the parameter α. At $\alpha = 14.0$, the PES is very rugged - the number of minima and barrier heights increase as the range is decreased [Doye & Wales, 1996]. Small values

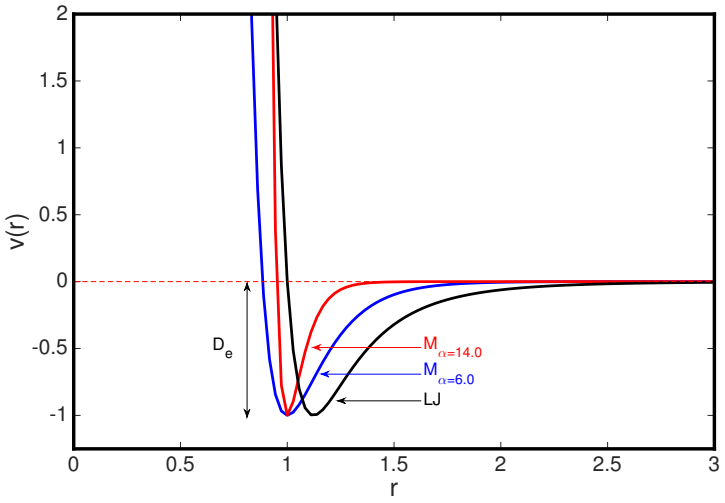


Figure 1.9: A graph of energy, $V(r)$, versus distance, r, under the Morse potential for two different range parameters, $\alpha = 6.0$ and $\alpha = 14.0$. At $\alpha = 6.0$, the Morse potential has the same curve at the bottom of the well as the LJ potential.

of α characterize long range potentials. As the range decreases the number of local minima increases for any given cluster size, which in turn makes the global optimization problem more difficult [Wales, 2003;Cassioli *et al.*, 2009]. Plots of energy of two atoms interacting within the Morse potential with $D_e = r_e = 1.0$ are shown in Figure 1.9. At $\alpha = 6$, the Morse potential has the same curvature at the bottom of the well as the LJ potential. Due to this, the most stable Morse clusters for this potential range are poly-tetrahedral icosahedral structures. At higher values of α, close-packed and decahedral structures begin to occur [Johnston, 2003;Doye & Wales, 1997]. The greater variety of stable structures found in Morse clusters across sizes provides a more rigorous testing ground for any global optimization methods.

1.4.4 Janus Particle Clusters

Janus particles, a term originally coined by Casagrande *et al* [Casagrande & Veyssie, 1988] in 1985, are named after an ancient Roman God depicted in Figure 1.10a. His two faces represent looking into the past and to the future, to the beginning and the end, to creation and destruction. This principle of ambivalence was the inspiration for referring to particles whose surfaces are made up of incompatible materials (e.g. hydrophobic and hydrophilic).

Applications

The distinct morphology and form of a Janus particle allows it to have some novel and even unique properties. There have been theoretical and experimental

(a) (b)

Figure 1.10: (a) The two-faced god Janus [Holistory, n.d.]. (b) Schematic view of a basic spherical Janus particle with sides A and B representing different physical or chemical properties.

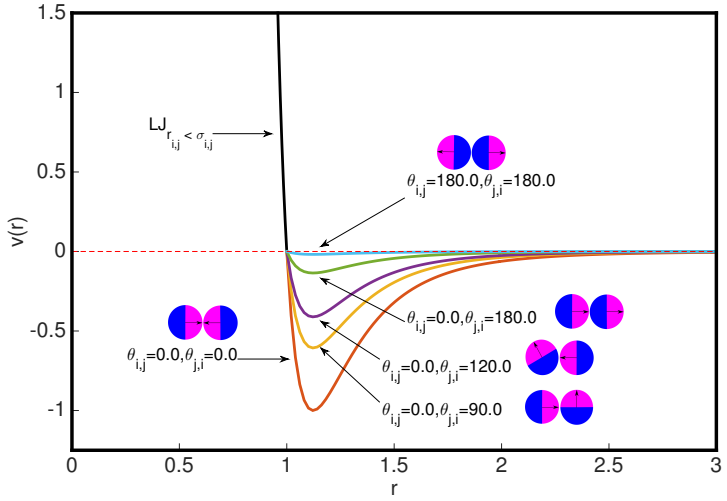


Figure 1.11: A graph of energy, $V(r)$, versus distance, r, under the Janus single-patch potential for different pairs of orientation measures.

investigations on their amphiphilic, magnetic, catalytic, optical and electrical properties [Hu *et al.*, 2012]. A few of the more recent studies on synthesis and applications are presented below.

The first potential application of Janus particles was demonstrated by Nisisako et al. [Nisisako *et al.*, 2006] to make switchable screens by placing a thin layer of spheres with white and black pigments between two electrodes. Upon changing the applied electric field, the particles orient their black sides to the anode and their white sides to the cathode. Thus the orientation and the color of the display can be changed by simply reversing the electric field. This method would allow the creation of very thin and environmentally friendly displays.

Magnetic Janus nanoparticles could become an effective tool in cancer treatment. In 2010, Janus nanoparticles synthesized from hydrophobic magnetic nanoparticles on one side and an amphiphilic block copolymer on the other side were used for imaging and magnetolytic therapy [Hu & Gao, 2010]. The magnetic side of the Janus nanoparticles responded well to external magnetic stimuli. The nanoparticles were quickly attached to the cell surfaces using a magnetic field. Magnetolytic therapy, which was described by the author as cancer cell therapeutics based on magnetically controlled mechanical forces, was achieved through magnetic field-modulated cell membrane damage. First, the nanoparticles were brought close in contact with the tumor cells, and then a spinning magnetic field was applied. After 15 minutes, the majority of the tumor cells were killed. Furthermore, a new type of cellular probe, called a nanocoral, combining cellular specific targeting and biomolecular sensing, was synthesized from Janus nanoparticles [Wu *et al.*, 2010]. A nanocoral is composed of polystyrene and gold hemispheres. The polysterene region was functionalized with antibodies that specifically attached to breast cancer cells. The gold region of the nanocoral surface was used for detecting and imaging. This makes the targeting and sensing to be engineered separately for a particular experiment. The polystyrene region could also be used as a carrier for drugs and other chemicals , making the nanocoral a possible multifunctional nanosensor.

Also in 2011, Janus nanoparticles were shown to be very efficient in the design of water-repellent textiles. Water-repellent fibers were prepared by coating polyethylene terephthalate fabric with amphiphilic spherical Janus nanoparticles [Synytska *et al.*, 2011]. The Janus particles bind with the hydrophilic reactive side of the textile surface, while the hydrophobic side is exposed to the environment, thus providing the water-repellent behavior.

Controlled manipulation of small volumes of liquids is of importance in miniaturized systems for chemical and biological applications. Contributing to this field, Zhao *et al* [Zhao *et al.*, 2013] developed a new kind of Janus particle

with multiplexed features including magnetic properties, which can control transportation and coalescence of liquid marbles, and an anisotropic color which can be employed for barcoding of the encapsulated liquid marbles. The particles were made to anchor at the air-water interface and acted as a highly flexible barrier for preventing coalescence of water droplets.

Kern-Frenkel Model for Patchy Colloids

For theoretical studies on patchy colloids, the Kern-Frenkel (KF) [Kern & Frenkel, 2003] potential has been a paradigmatic model for highly anisotropic interactions. This model consists of a hard-sphere potential and an orientation-dependent attractive part. Studies by Sciortino *et al* [Sciortino *et al.*, 2010], Giacometti *et al* [Giacometti *et al.*, 2012], Hong *et al* [Hong & Cacciuto, 2012], and Fantoni *et al* [Fantoni *et al.*, 2014] used this potential where hard-sphere potential is the isotropic square-well tail and the orientation-dependent term vanishes when the interparticle vector does not intersect attractive patches on each particle.

orientation-dependent term is an isotropic square-well tail. More precisely, the two-body Kern-Frenkel potential as presented in [Sciortino *et al.*, 2010] is:

$$u(r_{ij}) = u^{sw}(r_{ij}) f(\Omega_{ij}) \tag{1.12}$$

The square well term $u^{sw}(r_{ij})$ with a depth of u_0 is shown in Equation 1.13. The function $f(\Omega_{ij})$ (Equation 1.14) is an angular term which depends on the orientation of two interacting particles.

$$u^{sw}(r_{ij}) = \begin{cases} +\infty & r < \sigma \\ -u_0 & \sigma < r < \sigma + \Delta \\ 0 & \sigma + \Delta < r \end{cases} \tag{1.13}$$

$$f(\Omega_{ij}) = \begin{cases} 1 \quad \text{if} \begin{cases} \hat{r}_{ij} \cdot \hat{n}_i > cos\theta & \text{patch on particle i} \\ and & \\ \hat{r}_{ji} \cdot \hat{n}_j > cos\theta & \text{patch on particle j} \end{cases} \\ 0 \quad \text{else} \end{cases} \tag{1.14}$$

In this model, σ is the hard sphere diameter, θ is the semi-amplitude of the patch, $\hat{n}_i$ and $\hat{n}_j$ are the patch unit vectors of particles i and j, respectively, r_{ij} is the Euclidean distance, $\hat{r}_{ij}$ is the interparticle vector between particles i and j and Ω_{ij} is the orientation of the interacting particles.

Using the KF potential, Sciortino *et al* [Sciortino *et al.*, 2010] evaluated structural properties of the system by means of extremely long standard MC

simulations to reach a proper equilibrium. The simulations involved MC sweeps which attempted random translation and rotation for each particle. The location of the gas-liquid critical point was calculated using the Grand Canonical MC simulations. Their results show that the system organizes into a dispersion of orientationally ordered micelles and vesicles and can be approximated as a fluid of structures at low temperature. Three theoretical approaches, namely Monte Carlo techniques, integral equations and perturbation theory, have been discussed and contrasted against each other in the study of Giacometti *et al* [Giacometti *et al.*, 2012] to address the phase-diagram computation of patchy colloids under the KF model. In Hong *et al* [Hong & Cacciuto, 2012], numerical methods using simple models for dipolar and amphiphilic Janus particles were discussed. The KF model was also used for the amphiphilic Janus particles while pair potentials for the dipolar Janus particles were computed by summing the contribution of point charges which are decorated homogenously over its surface. Monte Carlo simulations were done to generate numerical results for the two cases. Fantoni *et al* [Fantoni *et al.*, 2014] constructed a cluster theory for the vapor of Janus fluid. Their approach considered the vapor phase as formed by clusters that are weakly interacting among each other and used simple fluid models to mimick their physical properties. And instead of using bulk simulation, the internal degrees of freedom of each clusters are obtained through a direct MC simulation of a single isolated cluster. A full description of the system, a combination of the two calculations, is then obtained using a procedure similar to the framework of simple fluids.

The hard sphere model can be difficult for methods other than Monte Carlo and does not lead itself to optimisation due to discontinous gradients.

Non Kern-Frenkel Potentials

Doye *et al* developed a different potential [Doye *et al.*, 2009] where the local structure can be more directly controlled. The potential is continuous a function of the orientation of the particles. The potential is shown in Equation 1.15. The particles in their model are "patchy" represented by a set of patch vectors where each patch can be attracted to any of the other patches.

The factor $V_{LJ}(r)$ is the isotropic LJ potential with σ_{LJ} as the distance at which two nonbonding particles can be at its closest. The orientational dependent term $V_{ang}(\hat{r}_{ij}, \Omega_i, \Omega_j)$ is presented in Equation 1.16. The parameter σ is the standard deviation of the Gaussian, $\hat{r}_{ij}$ is the interparticle vector connecting the centres of particles i and j, Ω_i is the orientation of particle i to the interparticle vector $\hat{r}_{ij}$, θ_{kji} is the angle between the patch vector k and the interparticle vector $\hat{r}_{ij}$. k_{min} is the patch that minimizes the angle θ. At $V_{ang} = 1$, the patches point

directly at each other and falls off as the patches deviate from further alignment.

$$V_{ij}(r_{ij}, \Omega_i, \Omega_j) = \begin{cases} V_{LJ}(r) & r < \sigma_{LJ} \\ V_{LJ}(r) V_{ang}(\hat{r}_{ij}, \Omega_i, \Omega_j) & r \geq \sigma_{LJ} \end{cases} \tag{1.15}$$

$$V_{ang}\left(\hat{\boldsymbol{r}}_{\boldsymbol{i},\boldsymbol{j}}, \Omega_i, \Omega_j\right) = exp\left(-\frac{\theta^2_{k_{min}ij}}{2\sigma^2}\right) exp\left(-\frac{\theta^2_{l_{min}ji}}{2\sigma^2}\right) \tag{1.16}$$

This model was first applied to crystallization in two dimensions. Using Monte Carlo cooling simulations for 4-, 5-, and 6-regularly shaped patches, and a patch size of $\pi/12$, crystallization was observed; albeit not quite perfectly, in the 4- and 6-patch simulations. In the 5-patch case, no overall crystalline order was observed; however particles do have one of two local environments. This local environment has been described as a tiling of two equilaterals and two squares. In their three-dimensional crystallization simulations, the 6-patch system was able to crystallize easily while the 4-patch system at best only generated partial crystallization. The 6-patch system exhibited a step-like decrease in energy on crystallization while the 4-patch system changed continuously.

Fejer *et al* [Fejer & Wales, 2015], on the other hand, predicted the global minima for tri-block Janus particles for small clusters of sizes $N = 2$ to 31, 72 and 100. The total energy for an N-system tri-block Janus cluster was defined as the sum of the Paramov-Yaliraki potential and an additional gravity term which acted as a force pulling each particle towards a soft repulsive wall in the xy plane. Unlike all numerical studies mentioned above, this study used a global optimization method called *basin-hopping* implemented in *GMIN* [Wales, n.d.] to generate their results. Their model allowed more than two neighbors to be formed per patch. With increased sedimentation effects, tetrahedral environments were progressively disfavoured while in the absence of sedimentation and where the Kagome lattice was bent, hollow structures were formed when size was $N = 100$. Ordering in bulk of soft triblock Janus particles was also investigated by Li *et al* [Li *et al.*, 2012]. The model they used was inspired by the soft particle model in dissipative particle dynamics and the Kern-Frenkel model. Configurations reported in their study did not include Kagome lattices but hexagonal columnar and body-centered tetragonal packing structures.

Perspective on Janus Particles Study

Based on statistics reported in *Janus Particle Synthesis, Self-Assembly and Applications* [Jiang & Granick, 2012] of publications on Janus particles from 1989 to 2011 found in the Web of Science database, 63% were dedicated to Synthesis and Characterisation, 16% to Assembly, 8% to Application, 6% to Review and only

7% to Theory and Simulation. Thinking beyond synthesis, a theoretical study would be the optimal approach. The limited study on the use of global optimization methods, such as nature-inspired algorithms, to find the global structures of Janus clusters is also worth looking into. This doctoral thesis aims to contribute to this effect by using a nature-inspired algorithm, specifically the Intelligent Water Drops Algorithm, to predict the lowest energies of Janus clusters.

1.5 Survey of Methods for Atomic Cluster Optimization

The succeeding sections present an extensive list of optimization algorithms that have been used over the past 30 years for atomic cluster optimization. Section 1.5.1 discusses the different algorithms and the principles behind them, while Sections 1.5.2 and 1.5.3 detail the relaxation algorithms and the different perturbation operators used to generate new clusters, respectively.

1.5.1 Algorithms

The *Northby Algorithm* [Northby, 1987] is one of the earliest and most successful algorithms for computing ground states of LJ clusters. The *Northby Algorithm* relied on the fact that it knows the underlying structure of the system it is optimising by setting up a lattice of atom sites with which initial configurations can be set up before lattice and local optimizations were done. The algorithm distinguished between "IC" and "FC" sublattice sites for pure LJ clusters. "IC" sublattice was described as all the sites that will comprise the outer shell of the next complete Mackay icosahedron while the *FC* sublattice consists of a smaller *IC* lattice enclosed by a layer of lattice points located at stacking fault positions of the *IC* lattice shell together with the vertex sites. The *Discrete-Continuous Algorithm* (DCA) [Maier *et al.*, 1992] used the same approach as the *Northby Algorithm* but it included a parallel implementation for both the lattice and local optimizations. The local optimization method used in Northby Algorithm was PARTAN [Petalas & Vrahatis, 2004] (See Section 1.5.2) while it was *L-BFGS* [Liu & Nocedal, 1989] (See Section 1.5.2) in DCA. Another algorithm based on Northby Algorithm and the Simulated Annealing is called *Parallel Two-Level Simulated Annealing Algorithm* (2L-SA) [Xue, 1994b]. The 2L-SA used *Northby Algorithm's* lattice optimization as the local minimization procedure while employing the idea of simulated annealing on accepting new configurations. It differed from the conventional simulated annealing in that it operated on two sequences of solutions: one sequence was the current feasible solution and the other was the locally minimized solution. The same author improved on this algorithm [Xue, 1994a] by

relaxing every lattice minimizer using a Truncated Newton algorithm and introduced a simple data structure which reduced the time complexity of the *Northby Algorithm* lattice optimization step.

A few more algorithms were based on simulated annealing : *Fast Annealing Evolutionary Algorithm* (FAEA) [Cai & Shao, 2002], *Sophisticated Fast Annealing Evolutionary Algorithm* (SFAEA) [Cai *et al.*, 2002b], *Parallel FAEA* (PFAEA) [Cai *et al.*, 2002a], and *Conformational Space Annealing Method* [Lee *et al.*, 2003]. The FAEA is a simulated annealing application to atomic clusters which used random structures as initial configurations and a very fast annealing schedule for the temperature. The generation of new clusters and the annealing schedule used based on the method called *Very Fast Simulated Re-annealing* by Ingber [Ingber, 1989] which permitted the temperature to decrease exponentially. The term "very fast" was derived from the fact that the annealing schedule used was faster than Cauchy annealing and much faster than Boltzmann annealing. FAEA did not mention the use of any local minimisation step. The SFAEA, on the other hand, was an improved version of PFAEA in that it included the local minimisation step L-BFGS, seeding of initial configurations, similarity checking for solution diversity and an extra step that displaces outer cluster atoms by spherical angles. The PFAEA is the parallel implementation of SFAEA but without the seeding of initial configurations component. Finally, the CSA unifies essential ideas from Simulated Annealing, GA and Monte Carlo with minimisation (MCM). It is similar to MCM in that only the phase space of local minima were considered. Similar to GA, CSA considered a population of configurations and generated a subset of configurations from these. And similar to SA, a diversity measure was introduced to diversify of the population. The diversity measure played the role of temperature in SA. CSA also employed diversity check and used random structures relaxed with L-BFGS as initial configurations.

A number of GA-based approaches to atomic cluster optimization have also been developed. Deaven *et al* [Deaven *et al.*, 1996] used an elitist GA with diversity check for atomic cluster optimization. This algorithm was originally applied to finding fullerence clusters up to size C_{60} [Deaven & Ho, 1995]. It used random initial configurations relaxed using an application of conjugate gradient minimization (CGM - See Section 1.5.2) of molecular dynamic quenching before applying the genetic algorithm operators. The crossover and mutation operators fitted for geometry optimization were employed. The diversity check ensured that a cluster in the population was within a diversity measure, δE, of the newly generated cluster after the application of GA operators. The diversity check step either replaced the member or the new cluster if the diversity measure was less than δE. A conjugate gradient minimization (CGM) method was used as a relaxation

method. In a study called *GA with Space-Fixed Coordinates* [Niesse & Mayne, 1996], which was inspired by a technique called *Deterministic-Stochastic Genetic Algorithm* (DS-GA) [Gregurick *et al.*, 1996], the manipulations to the clusters were recast in space-fixed Cartesian coordinates rather than on binary-coded internal coordinates (the case in DS-GA). In addition to random initial configurations, Niesse and Mayne implemented a seeding procedure adapted from Hoare and Pal [Hoare & Pal, 1971] where an N-atom cluster was seeded from a previously prepared optimal *(N-1)*-atom cluster. The nth atom was then randomly positioned on the surface of the optimed *(N-1)*-atom cluster. Furthermore, to prevent the clusters being identical, the seed $(N-1)$ atoms were randomly rotated about its centre of mass by generating three random Euler angles. GA operators fit for geometry optimization were used and relaxed with a CGM step. Another GA variant, called *Predatory GA* (PGA) [Manby *et al.*, 1998], borrowed on the analogy of predation in natural evolution and applied it to atomic cluster optimization. In addition to the standard GA steps including selection, crossover, and mutation, PGA included a diversity check that removes clusters from the population if they are closer than some threshold, ε_π, to the best cluster in the population found by standard GA. The standard GA generation of new clusters used in PGA was based on the GA implementaion for atomic cluster optimization by Deaven *et al* [Deaven *et al.*, 1996]. The GA implementation of Wolf and Landman [Wolf & Landman, 1998] added further modifications on the elitist GA with diversification approach of Deaven *et al* [Deaven *et al.*, 1996]. Specific modifications included the introduction of new GA operators and, in conjunction with random configurations, the seeding of initial population using structural motifs. For clusters in the size range $20 \leq N \leq 60$, the 13-atom and 15-atom icosahedron, cuboctahedron and decahedron were used as core seeds. For clusters in the size range $20 \leq N \leq 60$, the 55-atom icosahedron and cuboctahedron core seeds were used. CGMs were done on clusters after employing of the GA operators. Hartke [Hartke, 1999] developed a GA and called it *Phenotype Algorithm with Niches*. Similar to the GA approach of Deaven *et al* [Deaven *et al.*, 1996], genetic operators act directly on the clusters themselves in configuration space rather than on a string representation of the problem. Crossover and Mutation operators fitted for geometry optimization were used as well. In addition, niches, a form of diversity checking, were also employed to differentiate between clusters of varying geometries and was used to decide which geometries to include in the next GA generation. In "niching" configurations were rotated and oriented about its centre of mass in such a way that its projection will result into the maximum coverage of atoms onto a plane. Configurations with a certain projection measure, g, into the plane will only be carried on to the next generation if it is closer

than Δg to a geometry already selected. Random initial configurations relaxed by L-BFGS were also used in the study. Johnston and Roberts [Johnston, 2003] also developed the *Birmingham Cluster GA* which included the standard GA steps (parent selection, crossover and mutation) and a diversity checking step that was found to be useful in most of the GA variants presented here so far. L-BFGS was used as the relaxation algorithm and initial configurations were generated randomly for each x, y and z coordinate in the range $[0, N^{1/3}]$. A lattice-based GA [Xiang *et al.*, 2004] used three structural motifs, namely the icosahedron, Ino's decahedron and the complete octahedron, to jumpstart configurations as well. Lattice optimization similar to the *Northby Algorithm* was used and L-BFGS was employed as the relaxation algorithm. Roberts *et al* developed another elitist GA variant [Roberts *et al.*, 2000] for atomic cluster optimization which included standard GA steps including the tournament selection method, crossover and mutation operators and an elitist selection strategy. The operators used were fitted for atomic cluster optimizations and L-BFGS was used to subsequently relax clusters generated by the operators and the randomly initialized configurations. An evolutionary algorithm (EA), with generally similar GA steps as in the GA of Roberts [Roberts *et al.*, 2000] and the *Birmingham Cluster GA*, was also developed by Marques and Pereira [Marques & Pereira, 2010]. This was an extension of the same algorithm developed by Pereira *et al* [Pereira *et al.*, 2008] but was modified to enable the EA to deal with heterogenous clusters. Pullan developed a population based search algorithm (PBS) based on GA [Pullan, 2010] which incorporated the *Basin-hopping* algorithm in generating initial trial clusters and adapted some GA operators used in previous studies. A new directed mutation operator was also introduced. PBS follows a very similar initial cluster generation from that of Roberts *et al* and *Birmingham Cluster GA* but with further mutation. After locally minimising a group of 150 random trial solutions, optimisation is further applied to identify and repair surface and interior "defects" in the cluster. This elaborate population creation phase either generates an initial population which either contains the putative global minimum or contains clusters whose energies are reasonably close to the putative global minimum. The local minimization method used was L-BFGS. The most recent real-coded GA approach to atomic cluster optimization is called *Mixed Integer-Laplace Crossover Power Mutation* (MI-LXPM) [Deep *et al.*, 2011]. As opposed to the genetic operators used in Deaven *et al* [Deaven *et al.*, 1996] and Hartke [Hartke, 1999] which acted on the clusters themselves in configuration space, MI-LXPM acted on a string representation of the clusters to generate new configurations. MI-LXPM started with random initial configurations but there was no mention of the use of any relaxation method in MI-LXPM to relax the initial configurations nor the

clusters generated during the application of the genetic operators.

A class of algorithms which transforms the potential energy surface into a collection of interpenetrating staircases were also developed for atomic cluster optimization. The first of which is called *Basin-Hopping Algorithm* by Wales and Doye [Wales & Doye, 1997;Doye & Meyer, 2005, 2006]. Basin-hopping, a similar approach to the *Monte Carlo Minimisation* algorithm of Li and Scheraga [Li & Scheraga, 1987], transformed any point in the configuration space with the local minimum obtained by a geometry optimization on that point. In some runs of the algorithms, initial configurations were randomly generated where atoms were confined to a sphere of radius 5.5 reduced units. Seeds containing one more and one less atom were also used in some of the runs of Basin-hopping. For a seed that is 1 atom more (i.e. LJ_{N+1}), the atom with the highest pair energy was removed and 200 unrestricted Monte Carlo moves attempted. For a seed from a cluster with one atom less, the $N-1$ atoms are fixed for a hundred steps while angular moves are applied on the remaining atom which is randomly positioned on the surface of the core. The Polak-Ribiere variant of CGM was used as the geometry optimization method. Another basin-hopping type of algorithm incorporating a multi-start strategy, called *Monotonic Sequence Basin Hopping Algorithm* (MSBH), was also developed by Leary [Leary, 2000]. In MSBH, the Metropolis acceptance criterion was abandoned in favor of only accepting downhill steps. In order to avoid being trapped in a local minimum, an escape mechanism was done by simply restarting the sequence from a fresh initial random local minimum (multi-start strategy). There was no seeding used for the initial cluster configurations in MSBH and the local minimisation procedure used was PARTAN. Another variant of *Basin-hopping*, called *Basin-Hopping with Occasional Jumping* (BHOJ), was developed by Iwamatsu and Okabe [Iwamatsu & Okabe, 2004]. BHOJ transforms the complex energy landscape into a collection of basins (hopping), explores them by random Monte-Carlo moves using the Mteropolis acceptance criterion and occasionally accepts a configuration without rejection (jumping) when hopping stagnates into a local optima. Random initial configurations were used at the start of the algorithm and may have also used the Polak-Ribiere variant of CGM as in *Basin-hopping* as its relaxation method, although this is not explicit in the paper. A similar type of algorithm which also transformed the potential energy surface by modifying the objective function was developed by Locatelli and Schoen [Locatelli & Schoen, 2001]. The algorithm, called *Two-Phase Multi-start*, performed local minimisations on the configurations in two steps, first one on a modified objective function biased towards certain configurations and then used the result from this modified objective function to perform another local optimization on the actual objective function (the LJ potential in the case of their

study). Random initial configurations were used at the start of the algorithm however the generation was elaborate than usual. Every generated random site of an atom must be at least 0.5 units away from another atom already generated. If the distance is greater than a set threshold, R, the atom was shifted until its distance from at least one atom becomes equal to R. CGM was used to relax the clusters generated for both the modified and the actual objective functions. A population-based variant of MSBH, called *Population-Based Monotonic Basin-Hopping* (PBH), was later on developed by Cassioli *et al* [Cassioli *et al.*, 2009]. Single independent iterations of MSBH were done and results were compared to each individual in the population. A dissimilarity measure to keep the population sufficiently differentiated was employed and the criterion for substitution was derived from the algorithm *Conformational Space Annealing Method* of Lee *et al* [Lee *et al.*, 2003]. Initial configurations for PBH adopted two policies : The first policy was to use a seed from a database of results of previous runs of the algorithm, which does not necessarily have the same run parameters (e.g. number of atoms), and is not necessarily the best one in the database. For clusters with fewer than N atoms, more atoms were randomly added, whereas if the previous cluster had too many atoms, excess atoms were randomly culled. The second policy was the use of pure random intial configurations with the centre of each atom uniformly generated in a ball with radius dependent on the cluster size, N. Additional perturbation operators were also added in PBH to explore a suitable neighbourhood from each configuration and two different types of local optimizations were employed. The standard local optimization used L-BFGS under the original objective function and the other type of local optimization was two-phase. The two-phase local optimization started with locally optimizing under a modified objective function (Phase 1) which favoured certain cluster shapes under the context of the system that were tested (e.g. spherical, oblate or prolate) and then locally optimizing the result using the standard local optimization (Phase 2). Phase 1 of the two-phase local optimization was directly derived from the *Two-Phase Multistart* algorithm of Locatelli and Schoen [Locatelli & Schoen, 2003]. MSBH was further enhanced in an algorithm called *Two-Phase Monotonic Sequence Basin Hopping* (TP-MSBH) by Doye *et al* [Doye *et al.*, 2004]. In TP-MSBH, the two-phase search of *Two-Phase Multistart* by Locatelli and Schoen [Locatelli & Schoen, 2001] and MSBH by Leary [Leary, 2000] was combined to generate a new algorithm for atomic cluster optimization. A further enhancement on MSBH was developed by Grosso *et al* [Grosso *et al.*, 2007] and called it *Two-Phase Population-based MSBH* (2P-PBH). In 2P-PBH, the MSBH was implemented as a population-based approach with diversification check and the generation of new clusters were adapted from the two-phase search strat-

egy of Locatelli and Schoen [Locatelli & Schoen, 2001]. Kolossvary and Bowers developed a "move set" for an algorithm, called *Hidden-force Monte Carlo Algorithm* (HFA-MC) [Kolossvary & Bowers, 2010], which generalized the Monte Carlo search method of *Basin-Hopping* [Doye & Meyer, 2005]. The move set was described as an intricate multiplayer of the tug-of-war game. Teams (clusters) try to break an impasse (local minimum) by randomly assigning some players (atoms) to drop their ropes while other players (remaining atoms) are still tugging until a partial impasse (locally minimize the cluster excluding the dropouts) is reached. The dropouts are instructed to resume tugging again for all the teams to come to a new overall impasse (the locally miniminized drop-outs and non-drop-outs are subjected to local minimization again). HFA-MC used randomly generated initial configurations relaxed with L-BFGS. The local minimization used for the 'move 'set' was also L-BFGS. The most recent under the basin-hopping type of algorithm for atomic cluster optimization is the *Minima-hopping method* of Sicher *et al* [Sicher *et al.*, 2011]. The *Minima-hopping method* based its moves on molecular dynamics.

A number of nature-inspired algorithms based on swarm intelligence techniques were also tested on atomic cluster optimzation. The standard *Particle Swarm Optimization* (PSO) was also tested on atomic cluster optimization [Hodgson, 2002]. Since the standard PSO is designed to search for global optima in an n-dimensional search space of real numbers, each "particle" (i.e. atomic cluster) is characterized by a $3N$ real number corresponding to the x, y, and z positions of each atom where N is the number of atoms. The *Ant Colony Optimization* (ACO) algorithm, in conjuction with the *Monte-Carlo sampling method* for randomly placing scattering atom sites in $3D$ space, was also applied to atomic cluster optimization [Tomson & Greenwood, 2005]. The ACO for atomic cluster optimization of Tomson *et al* is similar to the standard ACO wherein "ants" gather components (i.e. atoms) of the solution progressively based on the strength of the atom connectivities (i.e. probability also includes dependency on potential energy between atoms) until the solution is complete. However, unlike the standard ACO, the ACO for atomic cluster optimization used a state transition rule that is *pseudo-random proportional* in selecting the next atom to move to. This type of state transition rule allows for either exploitation or exploration. A simple-hill climbing relaxation algorithm was used to relax final structures in ACO which moved each atom coordinate by 0.01 and replaced the original cluster only if the new cluster has a lower energy. Another variant of ACO developed for atomic cluster optimization was done by Raczynski and Gburski [Raczynski & Gburski, 2005]. They used the same *pseudo-random proportional rule* for selecting the next atom but their probabilities were based on an extra ant "activity". In their

ACO implementation, ants were allowed to temporarily displace the position of the other atoms one at a time. The probability of moving to that atom was then computed based on the pair potential energy of the ant's current atom position and the displaced atom. The displaced atoms were replaced back on their original positions and an ant moved to the next atom position associated with the highest probability. This ACO implementation did not mention any relaxation method but employed random initial configurations of clusters. A swarm intelligence based on simulating human social behaviour, called *Hybrid Social Cognitive Optimization Algorithm* (HSCOA) [Chen *et al.*, 2010], was also used for atomic cluster optimization. The algorithm is similar to that of PSO in that it considers both the population's and the cluster's performance to influence how much change (generation of a new cluster) is effected on that cluster. HSCOA introduced the concept of a "belief index", set to 1.0 at the start, which determines a cluster's "status in society" (i.e. the potential energy of the cluster). Belief indices were decreased if a cluster's society status did not improve and set back to 1.0 otherwise. It also employed L-BFGS to find lower energies and introduced a random displacement of a randomly selected atom in each cluster as a way of enhancing the global search capability of the algorithm and as a mechanism for escaping from the local optima.

There were other approaches for atomic cluster optimization which were generally different from the algorithms discussed so far. The first one is called the *Peeling Algorithm* by Barron *et al* [Barron *et al.*, 1997]. This algorithm imitates the act of peeling where optimization of an N-atom cluster is started from peeling atoms off (one at a time) of an $(N+1)$-atom cluster and then locally minimizing the "peeled" cluster. The local search procedure was not detailed in the paper although it suggested that any standard gradient routine can be used. Another approach is called the *Continuous Extremal Optimization* (CEO) of Zhou *et al* [Zhou *et al.*, 2005]. CEO draws upon the principle where the least fit elements are successively selected for adaptive changes. It used the probability function, $P(k) \sim k^{-\tau}$, where $k \geq 1$ is the rank of the element based on its fitness value and $\tau \geq 1$, to determine which element (atom in the case of atomic cluser optimization) was to be "changed". CEO used a locally minimized random initial structures at the start of the algorithm where atoms were placed within a spherical container whose radius varies based on the size of the cluster being tested. L-BFGS was employed as the relaxation algorithm. Tao *et al* [Tao *et al.*, 2011] developed *3OP* which made extensive use of three cluster operators, thus the name of the algorithm, which effected changes on the clusters in configurations space rather than on the string representation of the cluster coordinates. L-BFGS was employed to relax the random initial configurations and the

newly generated clusters after every application of each operator. A pure *Random Search Algorithm* was also implemented for atomic cluster optimization by Johnston [Johnston, 2003] as a way to compare the performance of *Birmingham Cluster GA*. The RSA was implemented by a simple generation of random initial clusters for which the x, y, and z coordinates of each particle were chosen using an identical method to that used in *Birmingham Cluster GA* and then minimised the energies of the clusters using the L-BFGS local minimisation routine.

1.5.2 Relaxation Methods

It is worth pointing out that most of the algorithms in Section 1.5.1 used some form of relaxation algorithm/local minimization step to find a lower energy in the nearby basin of the initial configurations. A few more algorithms used relaxation algorithms not just to relax the initial configurations but also the energies of the configurations generated within the iterations.

Conjugate Gradient Method (CGM) is a faster form of steepest descent. It has been shown to converge in n steps [Hestenes & Stiefel, 1953], where n is the number of decision variables. CGM is a local optimization technique which uses conjugate sets of directions rather than the fixed direction employed in steepest descent. The Polak-Ribiere variant of CGM was used in the *Basin-Hopping* algorithm [Wales & Doye, 1997]. Another variant called *Basin-hopping with Occasional Jumping* [Iwamatsu & Okabe, 2004], may have also used the same variant of CGM as its original variant, although this is not explicit in the paper. A GA-based algorithm by Wolf *et al* [Wolf & Landman, 1998] used CGM after the application of crossover and mutation operators to the clusters. *Two-phase Multistart* [Locatelli & Schoen, 2001] used CGM when locally minimizing configurations on the modified objective function and the original objective function. The *Peeling Algorithm* [Barron *et al.*, 1997] used a certain type of relaxation method (not detailed in the paper) for locally minimizing configurations. 2L-SA [Xue, 1994b] used the lattice-optimization technique of the *Northby Algorithm* as the local minimization procedure (detailed in next section).

The *Parallel tangent gradient search method* (PARTAN), on the other hand, is an improvement over the convergence of the simple gradient method [Petalas & Vrahatis, 2004]. PARTAN occasionally uses the difference between the current point and the point in 2 previous steps as a search direction instead of the gradient. It was used in the *Northby algorithm* [Northby, 1987] and MSBH [Leary, 2000] to locally minimize newly generated configurations generated during iterations. TP-MSBH [Doye *et al.*, 2004] and 2P-PBH [Grosso *et al.*, 2007], being variants of MSBH, used *PARTAN* to locally minimize generated clusters.

Limited-Memory BFGS (L-BFGS) [Liu & Nocedal, 1989], a more recent

form of all the relaxation methods mentioned in this section, is an optimization technique based on the quasi-Newton method. L-BFGS approximates the inverse of the Hessian matrix, denoted by H_k, by applying a number of BFGS updates to a diagonal matrix H_0 using information from the previous M steps. The L-BFGS algorithm is presented in Appendix A. The following algorithms used L-BFGS to subsequently relax the configurations generated within iterations or after the application of local operators: discrete lattice optimization section of the algorithm by Maier *et al* [Maier *et al.*, 1992], PFAEA [Cai *et al.*, 2002a], SFAEA [Cai *et al.*, 2002b], Lattice-Based GA [Xiang *et al.*, 2004], CEO [Zhou *et al.*, 2005], HSCOA [Chen *et al.*, 2010], HFA-MC [Kolossvary & Bowers, 2010], EA [Marques & Pereira, 2010], 3OP [Tao *et al.*, 2011] and the GA implementations of Roberts *et al* [Roberts *et al.*, 2000], *Birmingham Cluster GA* [Johnston, 2003], locality analysis of GA operators by Pereira *et al* [Pereira *et al.*, 2008] and PBS [Pullan, 2010]. HFA-MC, however, used a customized L-BFGS involving analytical second derivatives in a sparse Hessian representation. PBH [Cassioli *et al.*, 2009], in its two-phase local optimization, used L-BFGS in locally minimizing the clusters in both phases. The two-phase local optimization in PBH was similar to the *Two-Phase Multistart* algorithm except that L-BFGS was used as the local optimization method. PGA [Manby *et al.*, 1998] mentioned the use of a quasi-Newton type of miniminization for generated clusters, although the optimisation method was not detailed in the paper.

A very crude hill-climbing relaxation algorithm was used to relax final structures obtained by an ACO-based optimization of Tomson and Greenwood [Tomson & Greenwood, 2005]. All the cluster atom coordinates were displaced by a small measure ($\delta r = 0.01$) and replaced the original cluster only if the new one has a lesser energy. The paper on FAEA [Cai & Shao, 2002], on the other hand, did not mention any relaxation method used in any of the steps in the algorithm.

1.5.3 Perturbation Operators

A plethora of techniques have been used to generate new clusters for atomic cluster optimization. Some techniques acted on the string representation of the cluster coordinates while the rest on the clusters themselves in configuration space. The ruggedness of the potential energy surface of the LJ potential function requires algorithms to include these perturbation operators to overcome barriers and efficiently traverse the different local minima. Techniques also vary from exploitative to explorative. These techniques, which we also call *perturbation operators/techniques*, are detailed below.

In a *modified GA with spacefixed coordinates* [Niesse & Mayne, 1996], the perturbation operators used acted on the string representation of each cluster.

A cluster was represented as a $3N$-dimensional array of concatenated x, y, z coordinates. The algorithm was made to randomly choose 1 of the 5 operators in every iteration. The following perturbation techniques were used:

1. **Arithmetic Mean :** A mutation technique of two parent clusters where a new child cluster was produced by calculating the arithmetic mean of corresponding coordinates of parent clusters.

2. **Geometric Mean :** Another mutation technique by which a child is created by calculating the geometric mean of corresponding coordinates of parent clusters.

3. **Inversion :** This mutation works on a single parent where the array positions of a randomly selected portion of array elements are inverted.

4. **Two-point crossover :** This crossover operator produces two children clusters from two parent clusters. From a random cutpoint c, child cluster 1 acquires the coordinates from array position c to $3N$ of parent 1 in its first few sites and gets the first array position until $c-1$ from parent 2 as its last few sites. Child cluster 2 will get the parent 2 then parent 1 combination with similar coordinate assignment as in child cluster 1.

5. **N-point crossover :** This crossover operator also generates two children clusters from two parent clusters. Based on a random number, child 1 will either get parent 1 or 2's coordinate. If child 1 gets parent 1's coordinate for array position i, child 2 will get parent 2's coordinate for position i. This is done for each coordinate position in the array from 1 to $3N$.

PGA [Manby *et al.*, 1998] used two simple operators to generate new individuals. For an N-array string representation of a cluster and where an array element is a 3-tuple consisting of the x, y and z coordinates of an atom, one (mutation) operator in PGA simply replaced randomly chosen elements of the cluster with random values. Another (crossover) operator generated a child cluster by taking the first k elements of parent 1 and concatenating it with the elements from $k+1$ to N of parent 2, where N is the total number of atoms. In the MSBH [Leary, 2000], there was no explicit operator to perturb a cluster. New clusters were obtained by generating a random $3N$-dimensional coordinate vector with independent identically distributed components from a Gaussian distribution with mean = 0 and standard deviation = 0.21 and then subsequently locally minimized. Generation of new clusters from a uniform distribution was also used but this generated little difference to the performance of their algorithm. Similarly, the *Two-Phase Multistart* of Locatelli and Schoen [Locatelli & Schoen,

2001] did not particularly have any perturbation techniques as their algorithm always restarted from a randomly generated initial configuration and subjected these configurations to a two-phase local minimization using a modified objective function in Phase 1 and the original objective function in Phase 2. TP-MSBH [Doye *et al.*, 2004] and 2P-PBH [Grosso *et al.*, 2007] follow the same prinicple as the *Two-Phase Multistart* and the MSBH. The perturbation operator used in the FAEA [Cai & Shao, 2002] was based on the *Very Fast Annealing* algorithm of Ingber [Ingber, 1989]. For a cluster, x, represented as a $3N$-dimensional array, where N is the number of atoms, a new cluster is generated using the equation $x_i^{k+1} = x_i^k + y_i(B_i - A_i)$. The parameters A_i and B_i are the lower and upper bounds, respectively, of dimension i while y_i is a function with a value within the range $[-1, 1]$ and dependent on the annealing measure during the annealing time, k (See [Ingber, 1989] for details). Extensions of FAEA, SFAEA [Cai *et al.*, 2002b] and PFAEA [Cai *et al.*, 2002a] used the same perturbation technique. In the PSO application to atomic cluster optimization of Hodgson [Hodgson, 2002], generation of a new of a cluster, $x_i(t)$, was based on the standard PSO solution update equation, $x_i(t) = x_i(t-1) + v_i(t)$. The addend $v_i(t)$ is a function based on the difference of $x_i(t-1)$ to the best cluster in the population and to the cluster's previous best (see [Hodgson, 2002] for details). BHOJ [Iwamatsu & Okabe, 2004], on the other hand, used Monte-Carlo (MC) random walks to find new configurations. This generation of new configurations is a simultaneous displacement of all particles, in contrast to MC in classical statistical mechanics where usually a randomly chosen single particle is displaced. The HSCOA [Chen *et al.*, 2010] is similar to that of PSO in that it considered both the population's and the cluster's performance to influence how much change (generation of a new cluster) was effected on that cluster (see [Chen *et al.*, 2010] for details). In MI-LXPM [Deep *et al.*, 2011], two operators called Laplace Crossover and Power Mutation were used. Details of these operators will be presented in Section 2.7

In the lattice-based algorithm of *Northby Algorithm* [Northby, 1987], perturbation operators acted on the clusters themselves in configuration space. to generate a new cluster, an atom that is loosely-bound (has the highest energy) is culled out and replaced with a randomly placed tightly-binding atom (i.e. has the lowest contributed energy) on the surface. The GA-based algorithms DCA [Maier *et al.*, 1992], 2L-SA [Xue, 1994b] and Xiang *et al* [Xiang *et al.*, 2004], also used the same lattice-optimization technique used in the *Northby algorithm*. In a GA-based algorithm by Deaven and Ho [Deaven & Ho, 1995], first a random plane that passes through the center of mass of each of the two parent clusters was chosen then a child was produced by assembling the top half of the plane of parent 1 and lower half of the plane of parent 2. If the child does not con-

tain the correct number of atoms, the parts were translated an equal distance in opposite directions normal to the plane until the atom numbers were corrected. In the *Peeling Algorithm* [Barron *et al.*, 1997], to find a new configuration for a cluster of size N, an optimized cluster of size $N-1$ was "peeled". One at a time, an atom was peeled or taken out of the cluster of size $N-1$ and locally minimized. The configuration with the lowest energy among the relaxed N-atom configurations was retained. In the *Basin-Hopping* algorithm [Wales & Doye, 1997] for homonuclear clusters, the energy landscape was explored by displacing all coordinates by a random number in the range $[-1, 1]$ times the time step. As applied on heteronuclear clusters (clusters where an atom can have one of two types), the *Basin-hopping* algorithm [Doye & Meyer, 2005, 2006] included additional moves that allowed efficient exploration of the compositional aspect of heteronuclear clusters. Aside from random displacement of all atoms and rotations of low-energy atoms around the centre of mass, two type moves were added. The two moves involved swapping the positions of a type A and type B atom and switching the type of a single atom. The GA-based algorithm of Wolf *et al* [Wolf & Landman, 1998] used four types of mutations enumerated below:

1. Mutate a configuration by taking a random number of atoms and allowing them to randomly walk in configuration space. They concluded that This method did not dramatically change the efficiency of their GA-based algorithm.

2. Applied twinning mutation where a random plane passing through the centre of mass is chosen and one side of the cluster is rotated by a random angle. Compared to the random walk mutation, twinning provided better convergence time.

3. Two mutations, namely grow-and-etch and etch-and-grow, were developed which mimic certain physical processes operative in the formation of clusters and nanocrystallites.

    (a) Grow-and-Etch grows the cluster first by adding m randomly placed atoms on the surface of the cluster and gradually etching high-energy atoms while relaxing at every step until the cluster returns to the original number of atoms.

    (b) Etch-and-Grow is the exact opposite of Grow-and-Etch where etching is done first before slowing growing the cluster.

In the *Phenotype Algorithm with Niches* algorithm of Hartke [Hartke, 1999], in addition to the mating procedure used in Deaven and Ho, a modified version of the assembling of parent halves was also done. Instead of simply

assembling halves from different parents, the worst half of one parent by the best half of another parent. To determine the quality of cluster parts, each atom was assigned an individual contribution to the total potential energy which was calculated as half of the sum of the pair potential terms for this atom. The modified version offered emphasis on improving existing geometries while the Deaven and Ho procedure provided a more explorative character. Furthermore, mutation was also applied by randomly displacing a certain number of atoms by a distance determined by nearest neighboring atom and the approximate radius of the cluster. A directed mutation repair step was also invoked whenever there was no change in any one member of the population. The directed mutation operator used had some faint resemblance to the lattice-optimization procedure of the *Northby Algorithm*. The mutation used in the GA version of Roberts *et al* [Roberts *et al.*, 2000] was a simple reassignment of atom coordinates (within $[0, N^{1/3}]$) to approximately one-third of the atoms in the cluster before relaxing to the nearby minimum. As a crossover operator, a variant of the cut and splice operator [Deaven & Ho, 1995;Deaven *et al.*, 1996] was carried out where parents were initially randomly rotated before creating a cutting plane horizontally. The complementary fragments were then spliced together consequently generating a new cluster. This new cluster may or may not undergo mutation based on a randomly generated number. The resulting cluster was then subsequenly relaxed to its nearby local minimum. Both types of operators were also implemented in *Birmingham Cluster GA* [Johnston, 2003]. Additionally, two more mutation operators were used. One was twisting, where the upper half of a cluster was rotated about the z axis by a random angle relative to the bottom half. The second mutation was cluster replacement, where an entire cluster was removed and replaced by an entirely randomly generated cluster. The CSA [Lee *et al.*, 2003] used a very similar strategy with Deaven and Ho [Deaven & Ho, 1995]. After identifying the random planes from each parent cluster, a fraction of the atoms which are farthest from the plane in parent 1 will be replaced by the atoms in parent 2 which are also farthest from its own cut plane. In addition, a random rotation perpendicular to the cut plane was applied to put the new parts together. Another cluster generation under CSA was done by choosing an atom in the current cluster with the lowest coordination number and moving it to the neighbourhood of an atom with the second lowest coordination number. In CEO [Zhou *et al.*, 2005], a new cluster was generated by moving the atom which was least fit using a probability function, $P(k) \sim k^{-\tau}$, where $k \geq 1$ is the rank of the element based on its fitness value and $\tau \geq 1$. In the ACO for atomic cluster optimization of Tomson and Greenwood [Tomson & Greenwood, 2005], new clusters were only generated from the standard ACO method of building solutions (clusters) by progressively

acquiring the components (atoms) using a probability. Once the solution was complete, no perturbation was done on the clusters. The atom coordinates of the best cluster were kept and new random sites were introduced again for the next iteration of the algorithm. At this point, the ACO algorithm was restarted. The ACO implementation of Raczynski and Gburski [Raczynski & Gburski, 2005] also worked in a similar manner except that the probabilities were computed based on the pair potential energy of the current atom position of the ant and the (temporarily) displaced position of another atom. In this implementation, however, there was no mention if the next iteration will retain the coordinates of the best ant in the previous iteration. Three crossover and two mutation operators were tested in the hybrid algorithm of Pereira *et al* [Pereira *et al.*, 2008]. The operators used in this algorithm is a mix of techniques acting on a string representation of a cluster and on the spatial distribution of atoms in configuration space. The first crossover operator, cut and splice (C&S), was entirely similar to the one implemented by Roberts *et al.* The second crossover operator, *a generalized C&S (GenC&S)*, started by finding a random cutting plane (CP) on each of the parent clusters. A number of atoms from parent 1 which are closer to the CP were then copied to the child cluster. The remaining atoms were selected from parent 2 the same way but with the additional criteion that these remaining atoms were not too close to atoms already in the child cluster. These criteria will probably render the child cluster to have lesser number of atoms. When this happens, the remaining atoms were randomly positioned in the vicinity of the partially completed child cluster. The third crossover operator, called uniform crossover, simply randomly chooses atoms from both parents with equal probability. In this case, the operator does not act on the spatial distribution of atoms in configuration space but on the string representation of parents. The uniform crossover scanned the atoms of the parent clusters from atom 1 to N. At each position, an atom from either parent 1 or 2 was randomly chosen as the atom in that position for the child cluster. The first mutation operator was called sigma mutation. This mutation generated a new cluster, *child*, using the equation $child = parent + \sigma N(0,1)$, where *parent* represented the parent cluster, $N(0,1)$ a random number sampled from a standard Normal distribution and σ a parameter from the algorithm. Sigma mutation also acted on the string representation of the cluster. The other mutation operator, called flip mutation, assigned an entirely new random set of coordinates for each atom in the cluster. The flip mutation operator essentially replaced the entire cluster with an entirely new randomly generated cluster. PBH [Cassioli *et al.*, 2009] implemented 2 perturbation methods over the atom coordinates and 2 more over cluster compositions. The first coordinate perturbation operator randomly positioned an atom within an area (sphere with radius less

than half of the equilibrium distance between atoms of different types) around its current position. The second coordinate perturbation operator started by randomly selecting a direction originating from the geometric centre of the cluster and choosing the atom which has the maximum projection from that direction (essentially a 'surface' atom). The two compositional perturbation operators used in PBH were similar to the ones used in *Basin-Hopping Algorithm* as applied on heteronuclear clusters. PBS [Pullan, 2010] used two categories of perturbation operators: Coarse-grained search (CGS) and Fine-grained search (FGS). The role of the CGS operators was to move around the search space in large steps while FGS does otherwise. Two crossover operators were used in CGS:

1. Random crossover started with a random rotation of two clusters around the 3 major axes. A number of atoms were then selected and, using the ones most distant from the $x-y$ plane, swapped these by translating atoms using the most distant atom from each cluster as the basis for the translation.

2. Selective operator, on the other hand, combined "good" hemispheres from each cluster to generate the child clusters.

The three mutation operators in CGS, which affected a single cluster and were applied to all atoms in that cluster, are listed below :

1. Twist mutation rotated atoms about a random axis.

2. Perturb mutation randomly perturbed all atom positions.

3. Randomly generating entirely new clusters.

FGS employed mutation operators which included the Directed Optimisation operator, slide (cluster segments above and below a random plane were translated parellel to the plane), rotate (cluster segments above and below a random plane were rotated around an axis normal to the plane), and self-crossover. The goal of the Directed Optimization operator is to iteratively identify and repair surface and interior "defects" in clusters. Directed Optimization functions in one of three modes:

1. **Surface Repair :** This operator moves the atom with the lowest number of nearest neighbor (NN) atoms to the best adjacent vacant position near a target atom with the maximum (but less than 12) number of NN atoms. This is similar to the lattice optimization step of the *Northby Algorithm.*

2. **Interior Repair 1 :** This operator randomly selects an atom from the atom pair that are closest to each other and removes this from the cluster.

The remaining $N-1$ cluster is then locally minimized using L-BFGS and then adds the removed atom using the Surface Repair technique.

3. **Interior Repair 2 :** This operator randomly selects an atom from all atom pairs whose separation is less than 1.123, locally minimizes the remaining $N-1$ cluster using L-BFGS and then adds the removed atom back onto the cluster using the Surface Repair technique.

EA [Marques & Pereira, 2010] used crossover and mutation operators with certain probabilities. A variant of the Generalized Cut and Splice by [Pereira *et al.*, 2008] was used as a crossover operator in EA for heteronuclear clusters where a subsets of atoms from parent clusters were superimposed over each other to generate children clusters. Another operator, called Sigma mutation, was also used on a single atom by perturbing its coordinates with a random value sampled from a Gaussian distribution with mean = 0 and standard deviation $0.05(2^{1/6}N^{1/3}\sigma_{BB})$, where N is the number of atoms and σ_{BB} is a parameter of the potential energy function of the system being tested. Switch mutation operator, on the other hand, was also used to change the type of the atom. In HFA-MC [Kolossvary & Bowers, 2010], the tug-of-war analogy will be used here to describe the basic hidden-force algorithm move. If a multiteam tug-of-war reaches an impasse, significant opposing forces (the negative of the gradient) may be exerted upon cluster atoms. Disrupting this network of opposing forces will result in the collective rearrangement of cluster atoms. A disruption is for some teams (atoms) to simultaneously drop their ropes (drop their interactions from the potential energy function). The remaining teams (remaining atoms) then re-arrange due to their net nonzero tugging until they reach another impasse (locally minimize the non-dropout atoms). Then at this point the dropout atoms are allowed to resume tugging again until an overall impasse it reached (locally minimize the drop-outs with the recently locally minimized non-dropouts). The second local minimization was employed since the dropouts remained at their original positions, they may be brought in close proximity with the remaining atoms after the first local minimization step. This Switch mutation of EA was also adopted in 3OP [Tao *et al.*, 2011] but was renamed as FLIP Procedure. Two more perturbation operators were used in 3OP to generate new structures. The Knead procedure locates the top m high-energy atoms and randomly places each of them in the interior of the cluster which has a radius of 2 units from the centre of mass. Smooth procedure, on the other hand, finds the top s high-energy atoms and top t most vacant sites (e.g. free locations in the cluster's surface that would contribute a lower energy if an atom is placed there). The high-energy atoms are placed one at a time on a vacant site. Both procedures involves locally minimizing

generated clusters using L-BFGS before doing an elitist type of accepting new clusters. The *Minima-hopping method* of Sicher *et al* [Sicher *et al.*, 2011], as tested on heteronuclear clusters, used molecular dynamics moves with identity exchange followed by local geometry minimization.

1.6 Survey of Applications on Atomic Cluster Optimization

The algorithms for atomic cluster optimization discussed in the previous sections were applied on Lennard Jones, Binary Lennard Jones and Morse Clusters. In this section, their application on certain cluster sizes and their performances relative to other algorithms, where mentioned in their respective papers, are discussed.

1.6.1 Lennard Jones Clusters

The LJ clusters had been intensely studied as a test system for global optimization methods designed for configurational problems. The vast literature on the LJ system is a proof of that. Below is a discussion of the algorithms for which the LJ system was used as a test bed to validate their performances.

The *Northby Algorithm* [Northby, 1987] was applied to the LJ clusters and laid most of the global minima in the size range $N = 13$ to 147 than had previously been discovered. The DCA by Maier *et al* [Maier *et al.*, 1992] was able to rediscover best-known published configurations for $N \leq 150$ and provided new results on minimum energies for clusters up to $N = 1,000$ atoms using massively parallel processors.

Xue's 2L-SA [Xue, 1994b] was able to get satisfactory results for cluster sizes as large as $N = 100,000$ atoms. The improved version [Xue, 1994a] was applied on LJ sizes 13 to 150 and lower energy configurations than the results from the *Northby Algorithm* were found for $N = 65, 66, 75, 77$ and 134.

The GA-based algorithm of Deaven *et al* [Deaven *et al.*, 1996] was tested on clusters up to size $N = 100$ and the authors concluded that application of the mutation operators allowed discovery of the optimal configurations, including special geometries (e.g. $N = 38$). They were able to report lower energies for $LJ_{38}, LJ_{69}, LJ_{88}, LJ_{98}$ than had previously been published. They also reported lower energies for LJ_{65} and LJ_{76} although these had the same energies as the results from an earlier paper on an *improved Northby algorithm* by Xue [Xue, 1994a].

In a *modified GA with spacefixed coordinates* [Niesse & Mayne, 1996], the GA variant successfully rediscovered the global optima for clusters with up to size 55. Global minima for sizes 4 to 29 were rediscovered using unseeded clusters

while the rest were seeded. There was no mention which among the 5 operators (Arithmetic Mean, Geometric Mean, Inversion, Two-point Crossover, N-point Crossover) performed best, as the algorithm randomly chose 1 of the 5 operators in every iteration.

The *Peeling Algorithm* [Barron *et al.*, 1997] was also successfully applied to LJ clusters from $N = 55$ to $N = 147$. The algorithm was able to find novel structures for cluster sizes $N = 69, 78, 88, 98, 107, 113$ and 115. These structures with lower energies were initially compared to the *Northby Algorithm* results . They later found out that the results they generated for LJ_{88} and LJ_{98} coincided with the results of Deaven *et al* [Deaven *et al.*, 1996], a paper published a year earlier, which also beat the results of the *Northby Algorithm*.

In the *Basin-hopping* algorithm [Wales & Doye, 1997], the lowest known structures were located for all LJ clusters up to 110 atoms, including a number that had never been found before in unbiased searches. The new lowest energy structures were discovered for LJ_{69}, LJ_{78} and LJ_{107}. *Basin-hopipng* beat Deaven *et al* [Deaven & Ho, 1995] at this point for the lowest known energy for LJ_{69}. Preliminary runs for LJ_{192} and LJ_{201} were also performed at which a complete Marks decahedron and a complete truncated octahedron occured for these sizes. In the MSBH of Leary [Leary, 2000], all the putative global minima of LJ clusters up to size 110 were rediscovered and it was able to locate a configuration with a lower energy for LJ_{98} than had previously been reported at the time of publication. Leary compared it to the original *Basin-hopping* algortihm [Wales & Doye, 1997] and concluded that neither the original nor MSBH showed systematic relative advantage in terms of averga CPU time or number of local minimizations between encounters with the icosahedral global minima. Both were similarly successful for LJ_{38} but MSBH was much faster and generated more hit rates for the L_{75}, L_{76} and L_{77} decahedral clusters. MSBH was also attempted on LJ_{201} but was not able to reach the lowest energy structure reported in the study using the *Basin-hopipng* algorithm [Wales & Doye, 1997]. The *Basin-hopping* variant of Iwamatsu and Okabe, called BHOJ [Iwamatsu & Okabe, 2004], was tested on $LJ_{38,75-77,98,102-104}$ and for larger sizes $LJ_{107,185-187}$. BHOJ was found to performed better in terms of success rates compared with the original *Basin-hopping* [Wales & Doye, 1997] and MSBH [Leary, 2000].

The GA-based algorithm by Wolf and Landman [Wolf & Landman, 1998], a modification of the GA by Deaven *et al* [Deaven & Ho, 1995;Deaven *et al.*, 1996], was successfully used on LJ clusters of up to size 100. This paper claimed to have determined slightly lower energies for cluster sizes $N = 69$ and $N = 75 - 78$ than had previously been discovered although the CCD Energy Landscape Database [Wales *et al.*, n.d.] for LJ cluster global minima does not list them as proponents

for these sizes. Wolf and Landman also claimed their algorithm to have 3-4 fold faster convergence rate for the cluster size tested than the GA version of Deaven *et al* [Deaven *et al.*, 1996].

The *Phenotype Algorithm with Niches* [Hartke, 1999] successfully rediscovered the putative global minima for LJ clusters up to size 150. Hartke was the first one to have used niches through geometry projection onto a plane as a way to differentiate geometries and consequently diversify the next generation of population. The *Phenotype Algorithm with Niches* also claimed to have beaten previous studies in literature in terms of program scaling which only grows cubically with cluster size.

The *Two-Phase Multistart* of Locatelli and Schoen [Locatelli & Schoen, 2001] have been tested on LJ clusters up to size 80. The algorithm could not detect the putative global minima for many cases when $N > 60$ although the authors reported their algorithm was faster (up to two orders of magnitude) than had previously been reported for N values where the optimal structurew were known to be non-icosahedral ($N = 38, 75, 76, 77, 98, 102$).

The SA hybrid with GA of Cai *et al*, called FAEA [Cai & Shao, 2002], was also successfully tested on LJ clusters up to size 13. This was improved in an extension of the algorithm called [Cai *et al.*, 2002b]. SFAEA was able to successfully locate the known global minima for LJ clusters up to size 74 atoms. This improved version was different from FAEA in that it included a relaxation method, diversification, seeding of intial configurations and an extra move step for generating new clusters in addition to the move step of the original FAEA which was based on the study of Ingber [Ingber, 1989]. A further improvement to FAEA, called PFAEA [Cai *et al.*, 2002a], was developed and successfully located all the lowest known minima up to 116 atoms. The PFAEA was a parallel version of FAEA and generally similar to SFAEA but without the seeding of initial configurations.

The SA-GA-MC hybrid algortihm called CSA [Lee *et al.*, 2003] reproduced all published global minima configurations for $N \leq 201$ without using structural information of the known global energy minima. On the other hand, a lattice-based GA [Xiang *et al.*, 2004] which used structural motifs successfully rediscover the global optima for larger clusters of up to 309 atoms. It also predicted the optimal configurations from sizes 310 to 561 atoms which by far the lowest known energy configurations as reported in the CCD [Wales *et al.*, n.d.].

In CEO [Zhou *et al.*, 2005], the global minima for LJ clusters up to size 100 were succesfully rediscovered. Compared to their implementation of the random walk (RW) algorithm, their results showed that CEO performed much better in terms of average energy than RW for up to size 100 atoms. However, the deviation

between the CEO average energy results and the global minima became larger and larger as cluster size increased. They concluded that CEO may be a poor algorithm for very large clusters. In the RW algorithm, instead of the extremal optimization process, a randomly selected atom, regardless of fitness value, was displaced.

A few algorithms have also been developed and only been applied to small LJ clusters. In the PSO for atomic cluster optimization of Hodgson [Hodgson, 2002], global minima for up to size 15 atoms were successfully located except for LJ_{13}. In comparison with a simple GA, called *BasicGA*, implementation on the same cluster sizes showed that PSO beat BasicGA in all test sizes in finding lower energy configurations (even when PSO failed at LJ_{13}). The BasicGA results reported in the paper showed to have not hit the global minima of any of the cluster sizes tested. The ACO implementation for atomic cluster optimization by Raczynski and Gburski [Raczynski & Gburski, 2005] was tested on Argon cluster with 7 atoms interacting under the LJ potential model. The authors mentioned that the structure obtained from the ACO algorithm is not a guarantee that it corresponds to the global minimum potential energy of the said test instance but there was an evidence of increased level of condensation. The other ACO for atomic cluster optimization developed by Tomson and Greenwood [Tomson & Greenwood, 2005] was tested for the 7-atom Silicon cluster using the LJ potential model. The authors reported that their algorithm needed to be augmented with some sort of relaxation method in order to find good low energy structures especially when initial atom sites were randomly scattered in configuraiton space. Unless these random sites include the positions of the global minimum configuration, a relaxation method is indispensable. The HSCOA [Chen *et al.*, 2010] was tested for LJ cluster sizes 2-5, 7-9 and 12. HSCOA was able to locate the global minima and performed better in terms of success rates than the standard PSO [Kennedy & Eberhart, 1995] and SCOA [Cui & Cai, 2010]. MI-LXPM of Deep *et al* [Deep *et al.*, 2011] was tested on LJ clusters up to size 15. MI-LXPM was able to successfully locate all the putative global minima of the said test instances and beat earlier published results (based on GA, PSO and differential evolution) in terms of the number of functional evaluations. The authors specifically compared it to PSO and concluded that MI-LXPM proved to be better due to the perturbation operators used which does not necessarily try to only explore the points that have already been explored which is the case in PSO.

1.6.2 Binary Lennard Jones Clusters

The Binary LJ clusters offers a more challenging task than the one-component LJ system for any global optimization method for configuration problem. The

compositional aspect of the Binary LJ clusters is one challenge to consider aside from the large number of minima present due to the fact that homotops exists. The algorithms below, mostly with steps adapted from methods applied to the one-component LJ clusters, have attempted to find the lowest known energy structures for Binary LJ clusters.

The *Basin-hopping* algorithm, which was originally tested on the LJ clusters, was also applied to Binary LJ clusters [Doye & Meyer, 2005]. It was successfully applied to Binary LJ clusters up to size 100 atoms and established the first set of putative global minima for the size range. The PBH by Cassioli *et al* [Cassioli *et al.*, 2009] was tested on Binary LJ clusters of sizes $N = 30 - 100$ atoms. It was able to locate 95 improved solutions from the results of the *Basin-Hopping* algorithm. The EA of Marques and Pereira [Marques & Pereira, 2010] was successfully applied to Binary LJ clusters of sizes 10 to 50 atoms and generated a new putative global optima for $N = 38$ with $\sigma = 1.05$. This new global minima has not been updated in the CCD [Wales *et al.*, n.d.] as of this writing yet.

The HFA-MC of Kolossvary and Bowers [Kolossvary & Bowers, 2010], was successfully applied to Binary LJ clusters of sizes 90 to 100 atoms. A total of 17 new putative global minima were discovered, beating results published by PBH and *Basin-Hopping* [Doye & Meyer, 2005, 2006]. It was found out that in the previous known minima, the outer and inner clusters shells were comprosed of larger and smaller atoms. In 13 of the new minima in HFA-MC, the atoms were not as clearly separated by size. Two more recently developed algorithms were able to generate new low energy structures. The first one was the *Minima-hopping method* of Sicher *et al* [Sicher *et al.*, 2011]. The *Minima-hopping method* was successfully applied to Binary LJ clusters up to size 100 atoms and found 17 structures which are lower in energy than structures that had been listed in the CCD [Wales *et al.*, n.d.]. The 3OP algorithm [Tao *et al.*, 2011] was also tested on Binary LJ clusters up to size 100 and found 12 new global minima missed in the PBH [Cassioli *et al.*, 2009] and the *Basin-hopping* algorithm [Doye & Meyer, 2005].

1.6.3 Morse Clusters

The Morse clusters exhibit a different challenge to global optimization methods for configurational problems. For a relatively small cluster size ($N < 110$), the Morse system exhibits more diversified structural behviours than the LJ system making the potential energy landscape more rugged. This characteristic provides a global optimization algorithm an alternative test system to verify their effectiveness and possible generality.

The putative global minima for Morse clusters with up to size $N = 80$ for various range of interaction, $\alpha = 3, 6, 10, 14$ were first predicted using a molecular dynamic technique with the use of starting geometries that maximized the number of nearest neighbor contacts [Doye *et al.*, 1995]. The same authors published another study on Morse clusters using a different methodology. In the paper [Doye & Wales, 1997], molecular dynamics was dropped and the lattice-based starting geometries were augmented with basin hopping and an eigenvector-following algorithm to take steps directly between minima on the potential energy surface. In the same Morse cluster size range, they were able to find 302 new structures that are global minimum for some values of α.

The PGA of Manby *et al* [Manby *et al.*, 1998], aside from testing it on small Al_n and C_n clusters, was also successfully tested on small Morse clusters with sizes $N = 8 - 10$ for $\alpha = 6$. The GA implementation of Roberts *et al* [Roberts *et al.*, 2000;Johnston, 2003] located all previously published global minima with 19-50 atoms both for medium range ($\alpha = 6$) and short-range ($\alpha = 14$) Morse potentials. The GA successfully beat *Random Search Algorithm* (RSA) in a comparison of performance for cluster sizes $N = 20, 30, 40, 50$. The RSA was implemented by a simple generation of random initial clusters and then minimized using the L-BFGS local minimisation method.

The MSBH, initially applied to LJ clusters, [Leary, 2000] was also modified and tested on Morse clusters and called TP-MSBH [Doye *et al.*, 2004]. In the TP-MSBH, all the putative global minima of Morse clusters up to size 80 for interparticle force range $\alpha = 3, 6, 10$ and 14 had been detected. It also found new global minima for cluster sizes $N = 24, 25, 45, 48, 51$ and 78. The 2P-PBH [Grosso *et al.*, 2007], its parallel version on the other hand, was successfully tested for sizes between 41 and 80 atoms only for $\alpha = 14$. The 2P-PBH was able to generate new lower energy configurations for $N = 48, 64, 67$ and 72. The 2P-PBH, together with different dissimilarity measures employed increase population diversity, was also compared to TP-MSBH for selected, but known difficult, Morse cluster sizes $N = 30, 43, 55, 61$ and 79. The 2P-PBH, in most cases, resulted in reduced number of average local searches and higher success rates than 2P-MSBH.

A hybrid algorithm by Pereira *et al* [Pereira *et al.*, 2008] successfully located the global minima for Morse clusters ranging from 19 to 50 atoms where α was fixed to 14.0. Locality analysis was performed in this study for the various operators used (Cut and Splice, Generalized Cut and Splice, Uniform Crossover, Sigma Mutation and Flip Mutation) which showed that the Generalized Cut and Splice and the Uniform Crossover operators tend to generate children analogous to parents which were less dissimilar. Cut and Splice remarkably generated original descendants even when population diversity was low. Flip mutation has low

locality while sigma mutation was a more appropriate variation operator but a proper step value was suggested to be carefully chosen to ensure preservation of correlation between parents and child clusters.

The PBS of Pullan [Pullan, 2010] rediscovered all putative global minima for clusters in the range $5 \leq N \leq 80$ and $N = 147$ for $\alpha = 3, 6, 10, 14$. It also established the putative global minima for Morse clusters in the range $81 \leq N \leq 146$ for $\alpha = 14$. This does not, however, appear in the CCD [Wales *et al.*, n.d.] at the time of writing this thesis yet.

1.7 Motivation of the Study

The literature suggests that lattice-based search or introduction of seeds using optimized $N + 1$ or $N - 1$ clusters indeed allows for effective configurational optimization of atomic clusters under the LJ potential especially for larger clusters. These algorithms rely on the fact that there has been *a priori* information regarding the geometry of the system at the start of the algorithm before the optimization steps are applied. However efficient and effective they may be, the fact remains that different systems may not necessarily have the same underlying structures across cluster sizes for the most stable configurations (e.g. Morse clusters). Past literature also showed a good number of studies where using purely random initial configurations lead to finding the lowest energies of relatively large atomic clusters along with efficient local optimizers and exploration techniques though not necessarily for difficult-to-find clusters.

The different techniques for traversing the potential energy surface no doubt contributed to the successful rediscovering of the putative global minima for the different algorithms mentioned above but well designed or cleverly chosen starting points are also needed. They are indispensble steps in the algorithm which, to some extent, accelerates convergence to the desired lowest energy. Together with these perturbation techniques is the application of a relaxation method that allows for finding the nearby catchment basin.

A particular gap in these studies is a clever way to pose trial structures for subsequent manipulation with perturbation or optimisation cycles. This is the type of problem nature-inspired algorithms are good at. IWD is one such nature-inspired algorithm that uses a probabilistic technique on finding good paths or solutions of graph-like problems. Atomic cluster optimization fits into the type of problem IWD can solve as in its simplest sense involves finding a collection of points in space that would generate the best possible objective function value. Finally, the main idea of this thesis is to adopt IWD to provide an intelligent initialisation phase for global optimisation of atomic clusters. More specifically,

this thesis involves the following tasks :

1. Modify a stochastic nature-inspired technique, specifically the Intelligent Water Drops Algorithm (IWD), and apply it to find the global minima of a simple yet categorized as an NP-Hard problem of finding the most stable configuration (lowest energy) of an atomic cluster.

2. Adopt an originally discrete-type of optimization algorithm called, the Intelligent Water Drop Algorithm, for suitability to atomic cluster optimization.

3. Use of unbiased random initial configurations, as opposed to lattice or seeded configurations, to initialize iterations of the algorithm.

4. Implement various perturbation techniques and an efficient relaxation method to efficiently traverse the potential energy surface.

5. Validate the MIWD algorithm with benchmark atomic models which inherently present various degrees of difficulty for new global optimisation algorithms namely Lennard-Jones, Binary Lennard-Jones and Morse potential models.

6. Predict global minimum structures of Janus clusters by using MIWD with appropriate perturbation techniques.

Chapter 2

Methods

The methodologies used in this study mainly consist of the modified Intelligent Water Drops (MIWD) algorithm and perturbation operators. The algorithm in this study, which will be referred to as MIWD+PerOp/MIWD+CombiOp, aims to find the global optima of atomic clusters in a two-step process. The MIWD part is treated as Phase 1 while the application of the perturbation operators as Phase 2. Modifications to the original IWD algorithm, program structure, potential models, angular dependent term for the Janus system, parameter values settings, bounding volumes and perturbation operators for each system are detailed in their corresponding subsections.

2.1 Problem Representation

MIWD for atomic cluster optimization has the same problem representation in graph form (N, E) as in the TSP. It however varies in the number of nodes in N and number of edges in E. The number of nodes in the set N is equivalent to $Num\_Particles$ while the number of nodes in the set E $Num\_Particles \times (Num\_Particles - 1)$. $Num\_Particles$ represents the number of randomly scattered particles within a defined boundary called *bounding volumes* (details in Section 2.6) at the start of the algorithm. All $Num\_Particles$ particles are assigned an integer identifier from 1 to $Num\_Particles$ which will not change from the start until the end of the algorithm. The number of particles that an IWD must probabilistically select at the end of an iteration is N_C (where $N_C < Num\_Particles$).

IWD for atomic cluster optimization is represented as $IWD_j = \{pc_1, pc_2, ..., pc_i, ..., pc_{N_C}\}$ where $pc_i \in N$, $3N_C$ is the total number of decision variables, $j \in [1, N_{IWD}]$, N_{IWD} the total number of IWD agents in the population. Each pc_i is represented as a 3-dimensional Cartesian coordinates $pc_i = [x_i, y_i, z_i]^T$. An expanded representation of each IWD based on components of its Cartesian

coordinate values is presented below. The size of this expanded representation is $3N_C$.

$$IWD = \{pc_{1x}, pc_{1y}, pc_{1z}, pc_{2x}, pc_{2y}, pc_{2z}..., pc_{ix}, pc_{iy}, pc_{iz}..., pc_{N_C}, pc_{N_C y}, pc_{N_C z}\}$$

Furthermore, we interpret $soil(i,j)$ as the amount of soil between particle i to particle j and the probability $p_{i,j}^{IWD}$ as the probability of choosing the path to particle j from particle i. It is to be noted that in the atomic cluster optimization application of IWD, $soil(i,j) \neq soil(j,i)$. Consequently, $p_{i,j}^{IWD} \neq p_{j,i}^{IWD}$.

2.2 Modifications to IWD algorithm

There are four main modifications to the original IWD algorithm (Subsection 1.3.3) for the optimization of atomic clusters. New parameters not described in Chapter 1 Section 1.3 are detailed in this section.

1. The probability used to select which particle to include in an IWD has been modified from the original implementation (Equation 2.1) to the new one in Equation 2.2.

$$p_{i,j}^{IWD} = \frac{f\left(soil\left(i,j\right)\right)}{\sum_{k \notin V_C^{IWd}} f\left(soil\left(i,k\right)\right)} \tag{2.1}$$

$$p_{i,j}^{IWD} = \frac{f\left(soil\left(i,j\right)\right)\eta\left(i,j\right)}{\sum_{k \notin V_C^{IWd}} f\left(soil\left(i,k\right)\right)\eta\left(i,k\right)} \tag{2.2}$$

where

$$f\left(soil\left(i,j\right)\right) = \frac{1}{\varepsilon_s + g\left(soil\left(i,j\right)\right)}, \tag{2.3}$$

$$g\left(soil\left(i,j\right)\right) = \begin{cases} soil(i,j) & min_{l \in V_C^{IWD}} \geq 0.0 \\ soil(i,j) - min_{l \in V_C^{IWD}} soil(i,l) & otherwise \end{cases} \tag{2.4}$$

An extra factor, $\eta\left(i,j\right)$ (Equation 2.5), has been included in the original probability to represent the inverse of pairwise potential energy between atom i and atom j plus an addend of 2 to push the value to a positive number. The measure $V(\ldots)$ is the pairwise potential energy of the system being tested which can be any of the potential models described in Section 2.4. Lennard-Jones, Binary-Lennard Jones and Janus clusters in this study

are all based on the 12-6 potential for which the distance at which the potential function reaches its minimum value is $-\varepsilon$. The pair equilibrium separation r_{minLJ} of the 12-6 potential is at $2^{1/6}\sigma \sim 1.122\sigma$. In reduced units, $\sigma = \varepsilon = 1.0$, r_m will be 1.122 thus $-\varepsilon = -1.122$. For the Morse potential, the pair equilibrium separation $r_{minMorse}$ is at 1.0. A shift of +2.0 to the function value is thus a safe translation to put all values to positive. The addition of $\eta(i,j)$ is to account for the fact that $f(soil(i,j))$ is biased towards spatially close particle positions. Small distances between particle positions are not always desirable in atomic clusters. Thus, the pairwise potential energy will be able to provide appropriate spatial evaluation of the particles.

$$\eta(i,j) = \frac{1}{2 + V(\ldots)} \tag{2.5}$$

It is also possible to use the exponential function, or other appropriate functions, in lieu of the simple inverse function. Experiments in this study, however, only used the latter and tests where not done to compare performances of different functions.

2. The HUD, or the heuristic undesirability factor, is specifically chosen to fit atomic cluster optimization. This factor affects the amount of soil that is taken from a path and the amount of soil that is taken by an IWD as it passes through that path. The HUD that was used in this study is :

$$HUD(i,j) = 2 + V(\ldots) + \mu r_{i,j} + \beta \left(max\left(0, r_{i,j}^2 - D^2\right)\right)^2 \tag{2.6}$$

It is based on a previous study by Locatelli, *et al* [Locatelli & Schoen, 2001]. The second term is the potential energy of the two particles being tested. The addend 2 is to make sure that the second term is pushed to positive values the same way as it is being used in $\eta(i,j)$. The third term is the modified LJ potential that strongly penalizes the atoms as they move away from each other while the last (fourth) term has no influence on pairs of atoms close to each other but strongly penalizes atoms far away from each other. D is an underestimate of the diameter of the cluster while μ and β are the weights of the penalty. The values D, μ and β are all non-negative values. Pairs of atoms that are far apart from each other ($r_{i,j} \geq D$) generate less interaction and thus penalty is given to make this path less attractive.

3. Aside from identifying the best iteration agent or IWD, T_{IB}, the worst iteration agent, T_{IW}, is also recorded. Both information are used in updating

the soil content of certain paths. Equation 2.7, which uses information from T_{IB}, is the soil updating formula for paths belonging to the best agent and is adapted from the original implementation of IWD [Shah-Hosseini, 2007]. In addition, Equation 2.8 is the soil updating formula for paths belonging to the worst agent. The parameter $\rho \geq 0$ is a weight factor while N is the total number of atoms that the IWD must collect.

$$soil\,(i,j) = (1-\rho)soil\,(i,j) - \rho\frac{2soil^{IWD}}{N\,(N-1)} \tag{2.7}$$

where $(i,j) \in T_{IB}$

$$soil\,(i,j) = (1+\rho)soil\,(i,j) + \rho\frac{soil^{IWD}}{N-1} \tag{2.8}$$

where $(i,j) \in T_{IW}$

4. To further optimize results at the end of each run of the algorithm, the resulting IWDs are subjected to L-BFGS [LBF, n.d.], a limited-memory optimization algorithm in the family of quasi-Newton methods that approximates the Broyden–Fletcher–Goldfarb–Shanno algorithm using a restricted amount of computer memory.

2.3 Program Structure and Implementation

The flowchart of the population-based MIWD algorithm is shown in Figure 2.1. The details of the steps are outlined in the steps below. Steps 4 to 12 are referred to as Phase 1 (a schematic diagram of MIWD Phase 1 is shown in Figure 2.2) while step 15 is referred to as Phase 2. A construction step made by each IWD agent involves an acquisition of an atom site using the defined probability (Equation 2.2). A *journey* is the acquisition of an IWD of the required number of atoms, N_C, for the cluster. An *iteration* is defined as the completion of journeys of **all** N_{IWD} IWDs. An iteration is influenced by the previous iteration. A *run* is one pass (MIWD Phase 1 and Phase 2) of the entire algorithm. One run is independent of another run.

In the succeeding chapters, the algorithm will be referred to as MIWD+PerOp when in Phase 2 only a single perturbation operator is applied, else when two perturbation operators are applied, we refer to it as MIWD+CombiOp.

1. Initialize static parameters ($numIWD, a_v, b_v, c_v, \epsilon_v, a_s, b_s, c_s, \epsilon_s, \rho, p, \mu, \beta$, $initSoil$, $initVel$ and D) and the dynamic parameter $soil(i,j)$.

2. Scatter particle sites in 3D. The volume in which the particles are scattered is an operational variable of the method. This study used four different types of bounding volumes with which the particles were scattered. Two of the types are spheres, one with a fixed radius of 5.5 [Wales & Doye, 1997] (referred here as Wales bounding volume) and one with a variable radius [Cai *et al.*, 2002b] referred to here as Cai bounding volume.

$$rad = \frac{1}{2} + \left(\frac{3N}{4\pi\sqrt{2}}\right)^{\frac{1}{3}} \tag{2.9}$$

The remaining two bounding volumes, we refer to here as Hodgson and Chen bounding volumes, were cubic with side lengths of 4 and 3 units [Hodgson, 2002;Chen *et al.*, 2010], respectively. The cubes and spheres were all centered at the origin. More detail on the generation of sites is presented in Section 2.6.

3. Initialize the dynamic parameters vel^{IWD} and $soil^{IWD}$.

4. Generate each member, IWD, of the population. For each IWD, the set of visited particles is set to empty, $V_C^{IWD} = \{\}$. The first atom site of each IWD is set to a random atom chosen from among the generated sites in step 2 and accepted unconditionally except when it is similar to the first atom of other IWDs. The set V_C^{IWD} is then updated by adding the first atom.

5. Select the next atom, j, to include into based on the quality of connection it has with the recently added atom, i, using the Equation 2.2. The associated atom j for which the probability is highest will then be added to the IWD and the set V_C^{IWd} is updated to include the recently added atom.

6. Update the velocity of the IWD as it moves from atom i to to the recently added atom j using Equation 2.10.

$$velocity^{IWD}(t+1) = velocity^{IWD}(t) + \frac{a_v}{b_v + c_v soil^2(i,j)} \tag{2.10}$$

where $velocity^{IWD}(t)$ is the velocity of the IWD in generation t and a_v, b_v and c_v, all positive values, are the velocity update parameters. The parameter b_v is set to a very small value to avoid division by zero while a_v and c_v are weight factors that determine the influence of the soil content of the path on the velocity of the IWD. The parameter t is the iteration number. The value of t during the first iteration is 0.

7. Calculate the amount of soil that is taken from the path between atom i and j.

$$\triangle soil\,(i,j) = \frac{a_s}{b_s + c_s time\,(i,j;velocity^{IWD})} \tag{2.11}$$

where

$$time\left(i,j;velocity^{IWD}\right) = \frac{HUD\,(i,j)}{max\,(\varepsilon_v, velocity^{IWD})} \tag{2.12}$$

a_s, b_s and c_s, all positive values, are the velocity update parameters. The time, $\left(i,j;velocity^{IWD}\right)$, it takes for an IWD to pass from its current location i to the next location j is proportional to the heuristic undesirability factor $HUD\,(i,j)$ and inversely proportional to the velocity of the IWD $velocity^{IWD}$. As $HUD(i,j)$ is defined as a measure of *undesirability* for choosing a path from particle i to particle j, a *smaller* $HUD(i,j)$ value signifies a more *desirable* path. A desirable path (smaller $HUD(i,j)$) then means smaller $time\left(i,j;velocity^{IWD}\right)$ and consequently a bigger amount of soil taken from the path i to j. Moreover, those paths of the environment that are used (by virtue of the probability in Equation 2.2) by more IWDs will have less soil.

8. Update the soil content of the path from atom i to the recently added atom j and the soil added into the IWD using :

$$soil\,(i,j) = (1-\rho)soil\,(i,j) - \rho \triangle soil\,(i,j) \tag{2.13}$$

$$soil^{IWD} = soil^{IWD} + \triangle soil\,(i,j) \tag{2.14}$$

9. Repeat steps 3 to 8 for each IWD until all IWDs have completed their generation (i.e. each IWD have acquired the complete number of atoms).

10. Calculate the energy of each IWD using one of the cluster potentials in Section 2.4 for this iteration while taking note of the best, T_{IB} , and worst, T_{IW}, IWDs. The IWD is more preferred if it has the most negative total potential energy.

11. Update the soil contents of the paths belonging to both T_{IB} and T_{IW} using Equations 2.7 and 2.8, respectively. Updating the soil contents of these paths reinforces the desirability (for T_{IB}) or undesirability (for T_{IW}) or selecting them in the next generation or iteration.

12. Update total best IWD, T_{TB} (Equation 2.15), and total worst IWDs, T_{TW} (Equation 2.16), by comparing it with the current iteration's T_{IB} and T_{IW},

respectively. One iteration is complete at this point.

$$T_{TB} = \begin{cases} T_{TB} & T_{TB} \leq T_{IB} \\ T_{IB} & else \end{cases} \tag{2.15}$$

$$T_{TW} = \begin{cases} T_{TW} & T_{TW} \geq T_{IW} \\ T_{IW} & else \end{cases} \tag{2.16}$$

13. Repeat steps 3 to 12 until the maximum number of iterations is reached.

14. Repeat step 1 to 13 for N_{Runs} number of runs. This will generate N_{Runs} number of T_{TB}s. These T_{TB} will then be subjected to L-BFGS to allow it find a lower energy in the nearby basin.

15. The relaxed clusters are then subjected to further optimization by applying perturbation operators (detailed in Section 2.7) developed in this study and adapted from previous studies [Wolf & Landman, 1998;Deep *et al.*, 2011;Niesse & Mayne, 1996;Deaven *et al.*, 1996;Tao *et al.*, 2011].

2.4 Cluster Potential Energy

The energies of the clusters generated by the algorithm were evaluated by calculating the cluster potential energy using Equations 2.17, 2.18 and 2.19 for the Lennard Jones (homogenous and binary) clusters, Morse and Janus clusters, respectively.

$$V_{LJ}(r_{i,j}) = 4\varepsilon_{i,j} \sum_{i=1}^{n-1} \sum_{j>i}^{n} \left[\left(\frac{\sigma_{i,j}}{r_{i,j}} \right)^{12} - \left(\frac{\sigma_{i,j}}{r_{i,j}} \right)^{6} \right] \tag{2.17}$$

$$V_{Morse}(r_{i,j}) = \sum_{i=1}^{n-1} \sum_{j>i}^{n} \varepsilon \left[exp\left(\alpha\left(1 - r_{i,j}\right)\right) \left(exp\left(\alpha\left(1 - r_{i,j}\right)\right) - 2 \right) \right] \tag{2.18}$$

$$V_{Janus}\left(r_{i,j}, \theta_{ij}, \theta_{ji}\right) = \begin{cases} \sum\limits_{i=1}^{n-1} \sum\limits_{j>i}^{n} V_{LJ}\left(r_{i,j}\right) & r_{i,j} < \sigma_{i,j} \\ \sum\limits_{i=1}^{n-1} \sum\limits_{j>i}^{n} V_{LJ}\left(r_{i,j}\right) MV_{angle}\left(\hat{\boldsymbol{r}}_{\boldsymbol{i,j}}, \theta_{ij}, \theta_{ji}\right) & r_{i,j} \geq \sigma_{i,j} \end{cases} \tag{2.19}$$

where

$$MV_{angle}\left(\hat{\boldsymbol{r}}_{\boldsymbol{i,j}}, \theta_{ij}, \theta_{ji}\right) = f\left(\theta_{ij}\right) f\left(\theta_{ji}\right) \tag{2.20}$$

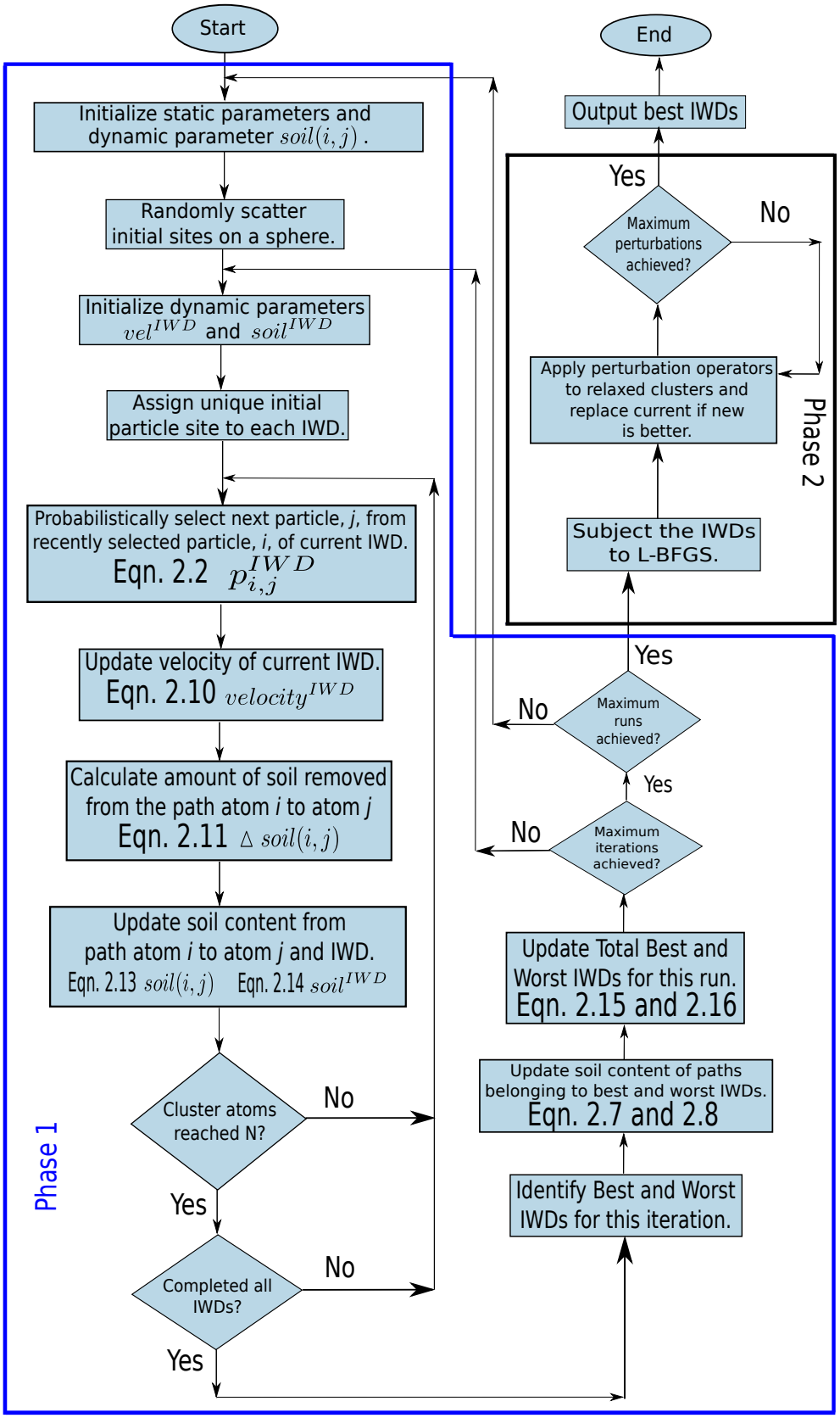
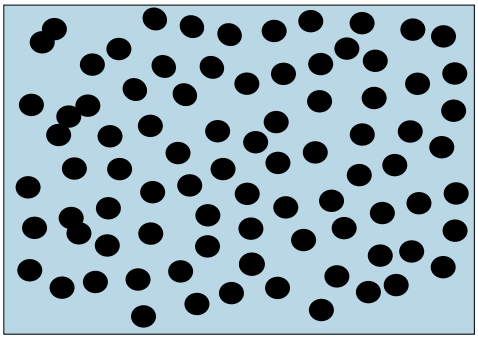
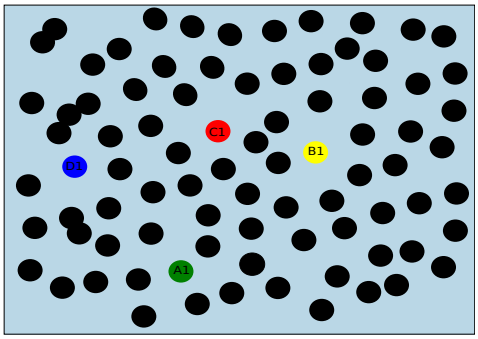


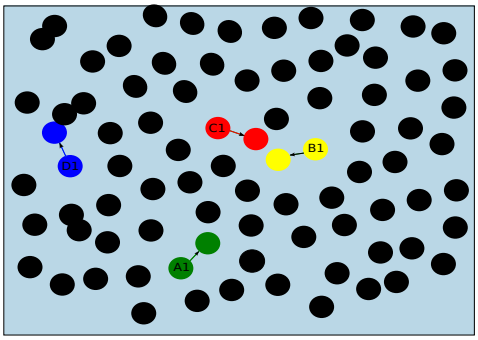
Figure 2.1: Flowchart of MIWD with Perturbation Operators for Atomic Cluster Optimization.



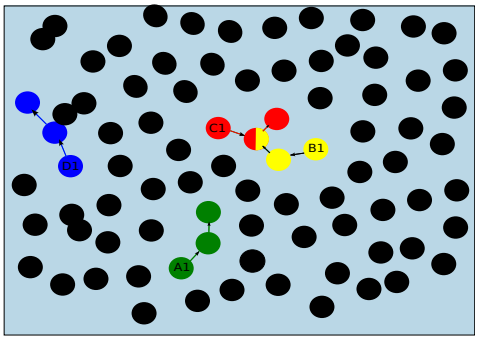
(a) MIWD Steps 1 and 2



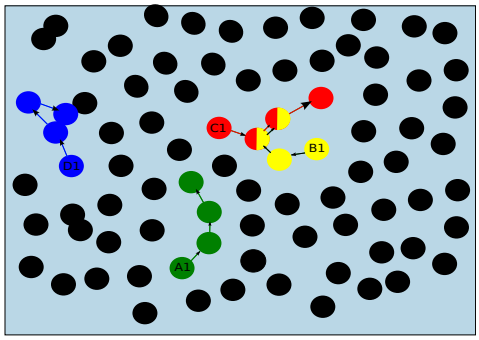
(b) MIWD Step 3



(c) MIWD Steps 4 to 9



(d) MIWD Steps 4 to 9



(e) MIWD Steps 4 to 13

Figure 2.2: One iteration of MIWD Phase 1 with 4 IWD agents (labelled $A1, B1, C1$ and $D1$) for a cluster requiring 4 atoms. (a) MIWD starts off with a random scattering of particles in a defined bounding volume in 3D configurational space and initialization of parameters. (b) IWD agents are randomly assigned a unique starting atom position. (c) - (d) IWD agents probabilistically choose the next atom to include in the cluster while updating its own properties ($soil^{IWD}$ and $velocity^{IWD}$). The arrows point to the next atom selected for inclusion to the cluster. In subfigures (d) and (e) it is probabilitistically possible for two IWD agents to choose the same atom in the course of the journey. As each IWD agent chooses an atom j to include into the cluster from atom i, the MIWD parameter $soil(i, j)$ gets updated (soil measure is lessened). (e) Upon completion of the journey by each IWD agent, the best IWD agent for this iteration and the total best IWD agent in all iterations, calculated using the potential energy function, will be recorded. The iteration is then restarted from Step 3 (subfigure (b)) with the IWD agents randomly placed on different starting atom positions. Succeeding iterations use the $soil(i, j)$ values from the previous iteration as a guide to new IWD agents.

$$f(\theta) = -exp\left(-\frac{\theta^2}{2\sigma^2}\right) + exp\left(-\frac{(\theta - 180)^2}{2\sigma^2}\right) \tag{2.21}$$

The orientational dependent term (Equation 9.2) is fully explained in Subsection 2.4.1. The following describes the parameters found in each of the models :

1. **LJ potential** : n is the number of particles in the cluster, $\varepsilon_{i,j}$ is the depth of the potential well, $r_{i,j}$ is the Euclidean distance between particle i and j and $\sigma_{i,j}$ is the finite distance where inter-particle potential is zero.

2. **Morse potential** : n is the number of particles in the cluster, ε is the depth of the potential well, $r_{i,j}$ is the Euclidean distance between particle i and j and α controls the width of the potential (the smaller the α, the larger the well)

3. **Janus potential** : n is the number of particles in the cluster, $r_{i,j}$ is the Euclidean distance between particle i and j, $\sigma_{i,j}$ is the finite distance where inter-particle potential is zero, $\hat{\boldsymbol{r}}_{\boldsymbol{i,j}}$ is the interparticle vector between particles i and j, θ (which could be either θ_{ij} or θ_{ji}) is the angle the patch vector ($\hat{p}_i$ or $\hat{p}_j$) makes with the interparticle vector (Figure 2.3) and σ controls the height of the Gaussian.

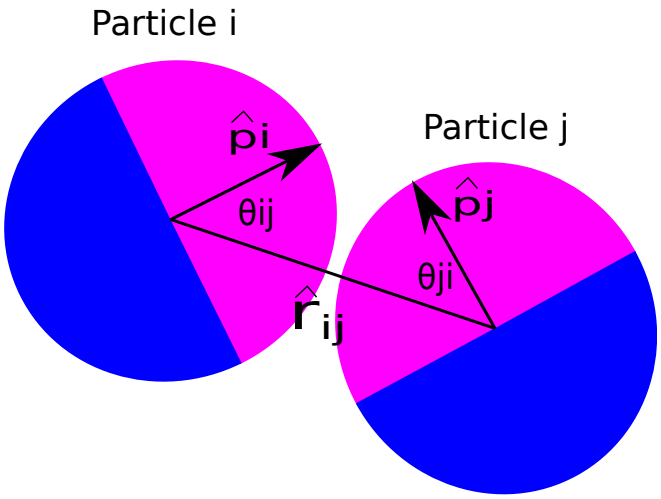
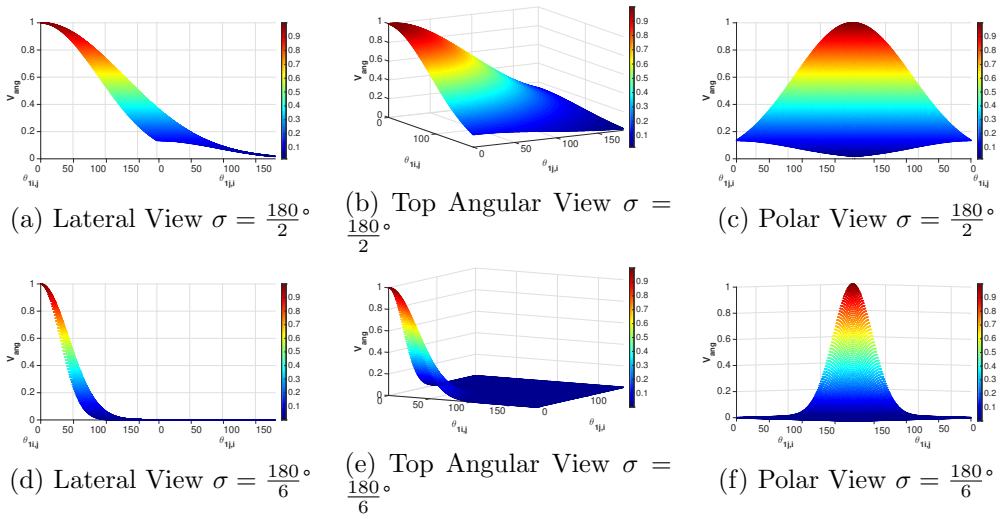


Figure 2.3: The geometry of interaction between two Janus particles.

2.4.1 Modified Angular term MV_{ang} for the Janus System

Equation 2.22 shows the orientational dependent term used in a study of patchy models by Doye and Allen [Doye *et al.*, 2009](detailed in section 1.4.4).



(a) Lateral View $\sigma = \frac{180}{2}°$ (b) Top Angular View $\sigma = \frac{180}{2}°$ (c) Polar View $\sigma = \frac{180}{2}°$

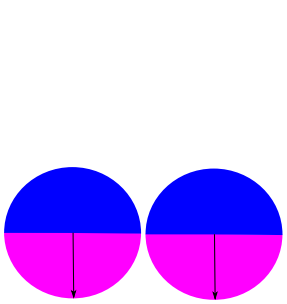
(d) Lateral View $\sigma = \frac{180}{6}°$ (e) Top Angular View $\sigma = \frac{180}{6}°$ (f) Polar View $\sigma = \frac{180}{6}°$

Figure 2.4: Orientation interaction of V_{ang} for $\sigma = \{\frac{180}{2}°, \frac{180}{6}°\}$, $0° \leq \theta_{1ij} \leq 180°$ and $0° \leq \theta_{1ji} \leq 180°$.

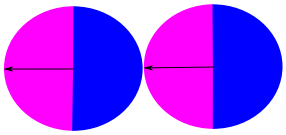
$$V_{ang}\left(\hat{\boldsymbol{r}}_{\boldsymbol{i,j}}, \Omega_i, \Omega_j\right) = exp\left(-\frac{\theta^2_{k_{min}ij}}{2\sigma^2}\right) exp\left(-\frac{\theta^2_{l_{min}ji}}{2\sigma^2}\right) \tag{2.22}$$

This patchy model allows the particles that are aligned along the interparticle vector or axis $\hat{\boldsymbol{r}}_{\boldsymbol{i,j}}$ to interact more strongly with each other. This is continuous as a function of the orientations of the particles. When patches are pointing directly at each other V_{ang} is equal to 1.0, but falls as patches deviate from the perfect alignment. Orientation interaction of V_{ang} (Figure 2.4) show that Equation 2.22 is insufficient to define the behaviour of a two-patched particle. Since V_{ang} was originally developed for single-site potential, its value when $\theta_{ji} = 180°$ and $\theta_{ij} = 180°$ (i.e. patch vectors are facing different directions) is zero or close to zero. This renders the entire pairwise potential to be zero or close to zero. However, for a two-patched Janus particle it is desired to define an orientation measure that captures "attraction" when interparticle vector or axis is directly pointing each other or directly opposite each other (Figure 2.5c) and reduced attraction or even repulsion when the particles misalign (Figures 2.5a, 2.5b, 2.5d and 2.5e). Thus, a different orientation term is needed for Janus particles.

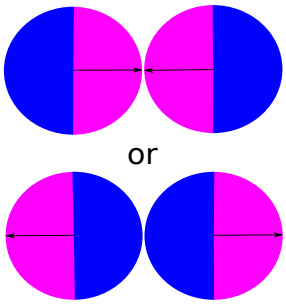
As already presented in section 2.4, the full form of the desired orientation term, MV_{angle}, is shown in Equation 2.23.



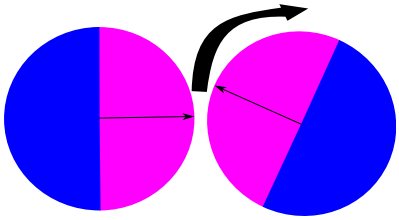
(a) Weak attraction, $cos^{-1}(\hat{r}_{i,j} \cdot \hat{p}_i) = 90.0$ and $cos^{-1}(\hat{r}_{j,i} \cdot \hat{p}_j) = 90.0$.



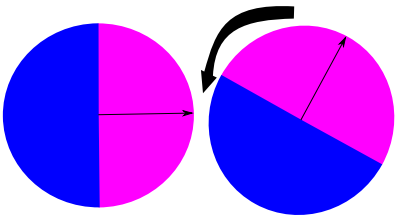
(b) Very strong repulsion where the orientation term must be at its minimum value, $cos^{-1}(\hat{r}_{i,j} \cdot \hat{p}_i) = 100.0$ and $cos^{-1}(\hat{r}_{j,i} \cdot \hat{p}_j) = 0.0$.



(c) Very strong attraction where the orientation term must be at its maximum value, $cos^{-1}(\hat{r}_{i,j} \cdot \hat{p}_i) = 0.0$ and $cos^{-1}(\hat{r}_{j,i} \cdot \hat{p}_j) = 0.0$ or $cos^{-1}(\hat{r}_{i,j} \cdot \hat{p}_i) = 180.0$ and $cos^{-1}(\hat{r}_{j,i} \cdot \hat{p}_j) = 180.0$.



(d) Orientation measure gradually falls off to minimum value as patch vectors points away from interparticle axis, $cos^{-1}(\hat{r}_{i,j} \cdot \hat{p}_i) = 0.0$ and $cos^{-1}(\hat{r}_{j,i} \cdot \hat{p}_j)$ is increasing from 0.0.



(e) Orientation measure gradually moves toward maximum value as patch vectors points towards each other, $cos^{-1}(\hat{r}_{i,j} \cdot \hat{p}_i) = 0.0$ and $cos^{-1}(\hat{r}_{j,i} \cdot \hat{p}_j)$ is decreasing to 0.0.

Figure 2.5: Schematic diagram of attraction and repulsion between patchy surfaces for two Janus particles. The vectors $\hat{r}_{i,j}$ and $\hat{p}_i$ refer to the interparticle axis and patch vector, respectively. Any orientation that falls in between the diagrams (a to c), the value of the orientation term falls between its minimum and maximum value (d and e).

$$
\begin{aligned}
MV_{angle}\left(\hat{\boldsymbol{r}}_{\boldsymbol{i,j}}, \theta_{ij}, \theta_{ji}\right) &= f\left(\theta_{ij}\right) f\left(\theta_{ji}\right) \\
&= \left[-exp\left(-\frac{{\theta_{ij}}^2}{2\sigma^2}\right) + exp\left(-\frac{\left(\theta_{ij}-180\right)^2}{2\sigma^2}\right)\right] \times \\
&\left[-exp\left(-\frac{{\theta_{ji}}^2}{2\sigma^2}\right) + exp\left(-\frac{\left(\theta_{ji}-180\right)^2}{2\sigma^2}\right)\right]
\end{aligned}
\tag{2.23}
$$

This equation will satisfy the following conditions:

1. MV_{ang} allows the energy to be repulsive when the inter-particle vector between particles does not intersect attractive surfaces and attractive otherwise.

2. MV_{ang}'s value falls off as the attractive patches deviate further from patch vector alignment or increases when attractive patch vectors start to align.

The first term of $f(\theta)$ is the attractive term while the second term is the repulsive term. As one patch vector, one which is on a perfect alignment with another patch vector, deviates from $\hat{\boldsymbol{r}}_{\boldsymbol{i,j}}$, $f(\theta)$ generally becomes less attractive. Numerically, the first term becomes less negative (or less attractive) while the second term becomes more positive (or more repulsive). The product $f(\theta_{ij})\, f(\theta_{ji})$ will generate an attractive MV_{ang} whenever each factor is negative. Relative interactions of orientation plotted against angular displacement for two prticles at contact are shown in Figure 2.7. Figures also show where MV_{ang} is attractive and repulsive. This is with the knowledge that $MVang$ is yet to be multiplied to $V_{LJ}(r_{ij})$ where the latter tends to move toward negative values when particles are attractive and towards positive values when particles are repulsive.

Following the schematic of Hong and co-worker [Hong & Cacciuto, 2012], figure 2.6 provides a simple sketch of the strength of repulsion and attraction between two Janus particles. In the first column are the two interacting particles in four different scenarios while the second column shows the overlapped surfaces of the two interacting particles. $S1$ indicates intersection of the same patch while $S2$ is the intersection of two different patches. A simple arithmetic of the difference in areas of $S1$ and $S2$ indicates repulsion and attraction. When $S2$ has a much larger surface area than $S1$, the interaction is repulsive and attractive otherwise.

Figure 2.8 shows the plot of Equation 2.19 for different pairs of $\{\theta_{ij}, \theta_{ji}\}$. At $r_{i,j} < \sigma_{i,j}$, the plot of energy is similar regardless of pairings of orientation measures. At $r_{i,j} \geq \sigma_{i,j}$, energy goes between the range of $[-\varepsilon, \varepsilon]$. The figures also show the behaviour of the energy as the interparticle vector passes through similar patches or different patches.

2.5 Parameter Values

Table 2.1 and 2.2 show the values used for the different parameters in MIWD and potential energy functions, respectively. Description of MIWD parameters can be found in the algorithm steps where they are used. Some parameter values were arbitrarily chosen while some were taken from previous literature where they were deemed optimal. There was no further attempt to optimize the parameters used in MIWD. In particular, the parameters in Table 2.1 do not change with cluster size.

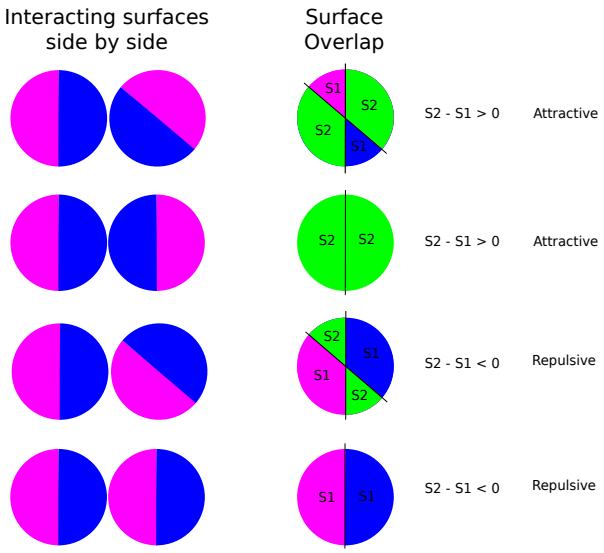
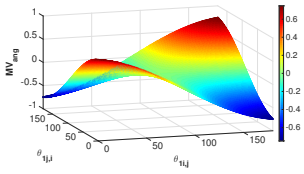
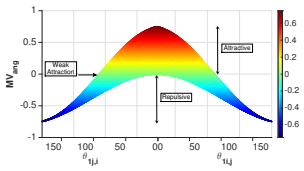


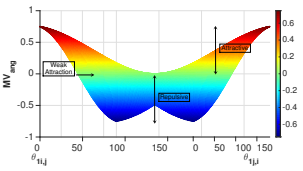
Figure 2.6: Sketch of strength of interaction between two Janus particles facing each other in different scenarios. The right column shows the surfaces on top of each other. $S1$ indicates overlapping of same surfaces from two particles while $S2$ are overlap of two different surfaces. Larger $S2$ surface areas shows repulsion and attraction otherwise.



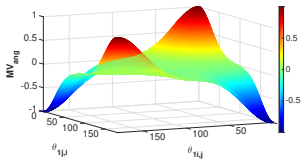
(a) Top Angular View $\sigma = \frac{180}{2}°$



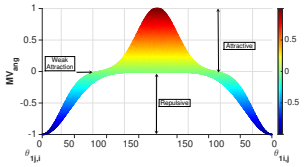
(b) Polar View $\sigma = \frac{180}{2}°$



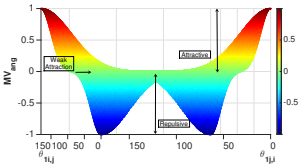
(c) Lateral View $\sigma = \frac{180}{2}°$



(d) Top Angular View $\sigma = \frac{180}{6}°$



(e) Polar View $\sigma = \frac{180}{6}°$

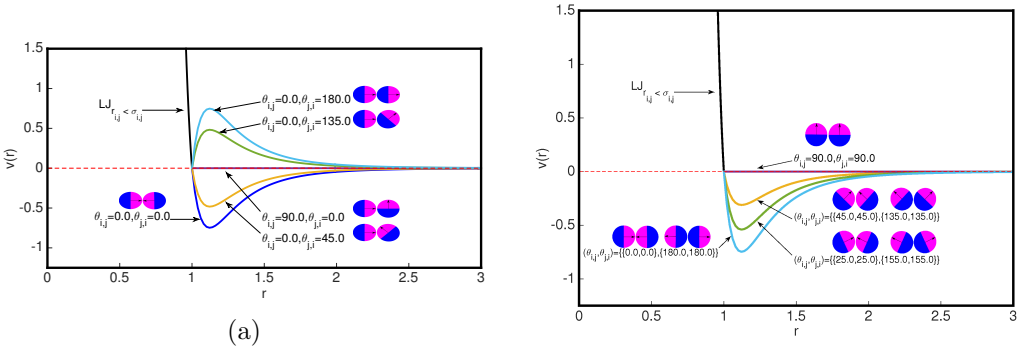


(f) Lateral View $\sigma = \frac{180}{6}°$

Figure 2.7: Orientation interaction of MV_{ang} for $\sigma = \{\frac{180}{2}°, \frac{180}{6}°\}$, $0° \leq \theta_{1ij} \leq 180°$ and $0° \leq \theta_{1ji} \leq 180°$.

2.6 Bounding Volumes

This study tested 2 types of bounding volumes with which to scatter the random initial sites for Phase 1 of the algorithm. The total number of sites is equivalent to $Num\_Particles$ and is set to $m \times N$ where N is the size of the cluster being tested and m is a positive integer. Apart from the preliminary tests on bounding volumes and perturbation operators, all final runs of all the test instances in this



(a) (b)

Figure 2.8: Plots of energies under the modified Janus Potential for different pairs of orientation measures. (a) Interparticle vector starts off with intersecting similar patch on both particles at $\{\theta_{i,j}, \theta_{j,i}\} = \{0.0°, 0.0°\}$ (bottom plot) to different patches at $\{\theta_{i,j}, \theta_{j,i}\} = \{0.0°, 180.0°\}$ (topmost plot). (b) Interparticle vector starts off with energy boundary between repulsion and attraction at $\{\theta_{i,j}, \theta_{j,i}\} = \{90.0°, 90.0°\}$ to gradual increase in attraction up to its maximum attraction at $\{\theta_{i,j}, \theta_{j,i}\} = \{\{0.0°, 0.0°\}, \{180.0°, 180.0°\}\}$.

study used $m \times N \sim 500$ random initial particle sites. Each type of bounding volume has two sizes.

1. Cubic (Figure 2.9a). Two sizes of cubes are tested in this study: 4 units and 3 units. We refer to the cube with side length of 4 units as Chen [Chen *et al.*, 2010] and the cube with length of 3 units as Hodgson [Hodgson, 2002]. The cubes are centered at the origin.

2. Spherical (Figure 2.9b). Two diameter sizes of spheres are tested in this study: 5.5 units and a cluster size-dependent radius, rad_{Cai} (Equation 2.24). We refer to the fixed sphere size as Wales [Wales & Doye, 1997] and the variable sized sphere as Cai [Cai *et al.*, 2002b]. The spheres are also centered at the origin. The pair equilibrium separator r_e is set to 1.0 [Chen *et al.*, 2010].

$$rad_{Cai} = r_e \left[\frac{1}{2} + \left(\frac{3N}{4\pi\sqrt{2}} \right)^{1/3} \right] \tag{2.24}$$

2.6.1 Initial Particle Positions for the Lennard Jones and Morse systems

All particles for the LJ and Morse systems were randomly generated in Cartesian coordinates within the defined bounding volumes detailed above. Since there are

Table 2.1: Parameter values for MIWD.

Parameter	Settings	Taken From
$a_v, b_v, c_v, \epsilon_v$	$\{1.0, 0.001, 0.01, 0.001\}$	This study
$a_s, b_s, c_s, \epsilon_s$	$\{1.0, 0.001, 0.01, 0.001\}$	This study
initSoil	1000.0	Taken from Shah-Hosseini [Shah-Hosseini, 2007]
initVel	0.0	This study
ρ	0.15	This study
p	4	Taken from Locatelli,*et al*[Locatelli & Schoen, 2001]
μ	0.50	This study
β	0.50	This study
D	4	Taken from Locatelli,*et al*[Locatelli & Schoen, 2001]

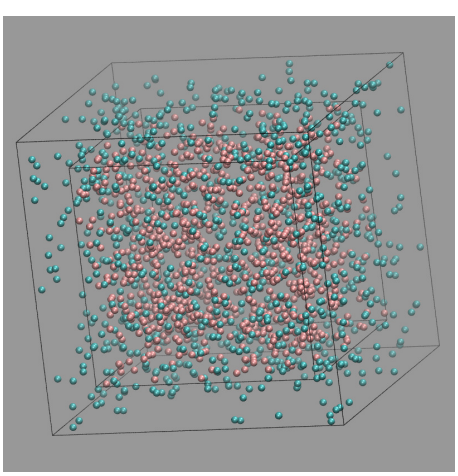
Table 2.2: Parameter Values for the different potential models.

System	Parameter	Value
Lennard	$\epsilon_{i,j}$w	1.0
Jones	$\sigma_{i,j}$	1.0
Binary	$\epsilon_{i,j}$	1.0
Lennard	$\sigma_{B,B}$	$\{1, 05, 1.10, 1.15, 1.20, 1.25, 1.30\}$
Jones	$\sigma_{A,A}$	1.0
	$\sigma_{A,B} = \sigma_{B,A}$	$(\sigma_{A,A} + \sigma_{B,B})/2$
Morse	α	$\{6, 14\}$
	ε	1.0
Janus	$\epsilon_{i,j}$	1.0
(LJ term)	$\sigma_{i,j}$	1.0

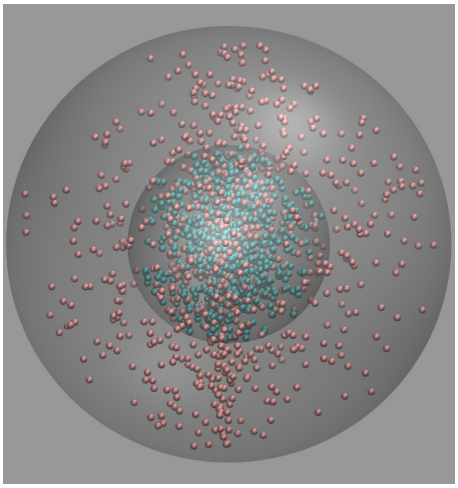
no additional information needed for LJ and Morse clusters, it was sufficient to only generate the x, y and z coordinates for each particle.

2.6.2 Initial Particle Positions and Types for the Binary LJ system

Each particle in Binary LJ was randomly generated in Cartesian Coordinates within the defined bounding volume similar to LJ and Morse systems. Since each particle in a binary LJ system could be one of two types, another random number, t, was generated to assign this information. The condition used to determine if the ith particle is type A or B is presented in Equation 2.25. Figure 2.10a shows randomly generated Binary LJ particles differentiated by type A (salmon) and B (navy blue) particle type.



(a) Cubic Bounding Volumes



(b) Spherical Bounding Volumes

Figure 2.9: Plots display the $m \times N$ random initial sites for each of the bounding volumes when $N = 50$ with each type of bouding volume visualized relative to each other. The value of m in the plots is set to 20. (a) Chen in navy blue while Hodgson is in salmon. (b) Wales in salmon while Cai is in navy blue.

$$type(i) = \begin{cases} A & t > 0.50 \\ B & otherwise \end{cases} \tag{2.25}$$

where i is from 1 to $m \times N$.

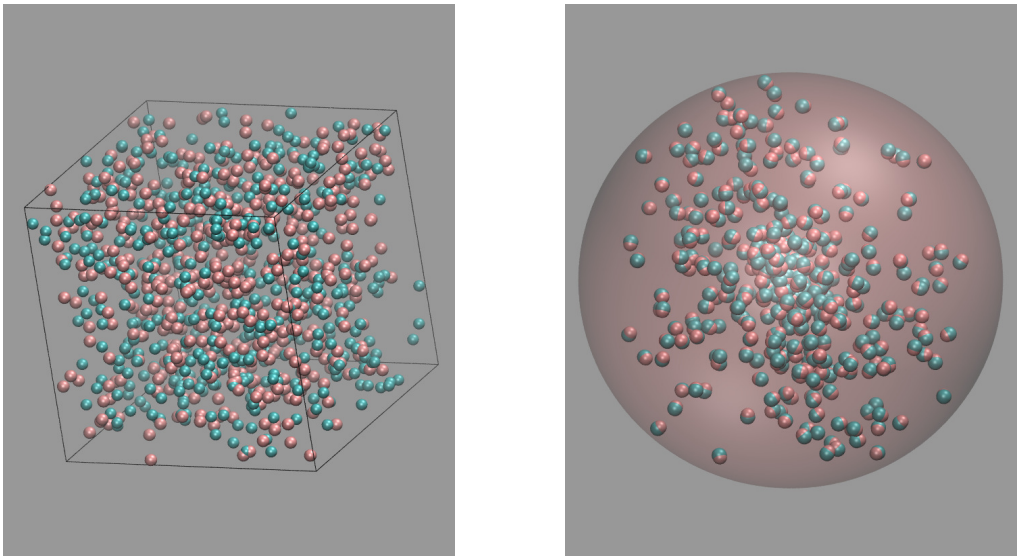
2.6.3 Initial Particle Position and Orientation for the Janus system

Cartesian coordinates of the positions of the initial particles for the Janus system were randomly generated first. The positions were bounded by the Wales bounding volume.

For the orientation of each particle, a random vector, $\hat{\mathbf{v}}_\mathbf{1}$, was then generated from the centre of the particle. This vector defines the center of one hemisphere of the particle. From here it was straightforward to generate another vector, $\hat{\mathbf{v}}_\mathbf{2} = -\hat{\mathbf{v}}_\mathbf{1}$, which defined the center of the second hemisphere. These two vectors consequently define the orientation of the particle. Figure 2.10b shows 100 randomly generated Janus particles with random positions and orientation.

2.7 Perturbation Operators

The following tables describe the perturbation operators used in each of the systems in this study and in the studies from which they were obtained.



(a) (b)

Figure 2.10: Sample random particles for (a) Binary Lennard Jones and (b) Janus clusters.

2.7.1 Operators for Lennard Jones Clusters

Ten perturbation operators where tested for the Lennard Jones system.Two operators, namely *Power Mutation* and *Laplace Crossover*, were based on the study by Deep and co-workers [Deep *et al.*, 2011]. Another five operators, namely *Inversion*, *Geometric Mean*, *Arithmetic Mean*, *2-point Crossover*, and *N-point crossover*, were taken from study of Niesse and Mayne [Niesse & Mayne, 1996]. The last three operators, namely *Twinning*, *Etch and Grow*, and *Grow and Etch*, were based on a paper by Wolf, *el al* [Wolf & Landman, 1998]. Details of these operators are presented in Algorithms 1 to 10.

In most of the perturbation operators, we use the expanded representation of IWD as a vector of $3N_C$ components representing the cluster configuration, where N_C is the number of particles in the cluster. For an IWD called ω, then its expanded representation is :

$$\omega = \{\omega_1, \omega_2, ..., \omega_{3i-2}, \omega_{3i-1}, \omega_{3i}, ..., \omega_{3N_C-1}, \omega_{3N_C}\}$$

where a triple $(\omega_{3i-2}, \omega_{3i-1}, \omega_{3i})$, for $i = 1, ..., N_C$, is a particle representation of its x, y and z Cartersian coordinates, respectively.

For the operators *Twinning*, *Etch and Grow* and *Grow and Etch*, generation of new IWDs are done on the geometry of the cluster rather than direclty on the $3N$ string representation of the IWD. For these operators, we also use the term *cluster* to represent the IWD.

The parameter values associated with each operator, if any, are also stated in each subsection. The values were chosen arbitrarily however no tests were done to tune them.

Power Mutation

This operator displaces all particles belonging to each IWD agent in the population. The degree of displacement is dependent on a power function and the upper and lower bounds of the components. All IWDs in the population will undergo Inversion. Pseudo-code of Power Mutation is shown in Algorithm 1.

Algorithm 1 Pseudo-code of Power Mutation

---

1. For each IWD $\overline{x}$ in the population

   (a) Define a power function $s = (s_1)^p$ where s_1 is a value taken from a uniform distribution $U(0,1)$ and p, the degree of mutation, is set to 1.0. Set x^l and x^u as the lower and upper bounds of the components, respectively. The bounds are dependent on the boundary volumes used during the scattering of particles sites.

   (b) Generate a random number $r \in U(0,1)$.

   (c) Create a mutated IWD $\hat{x}$ from the original IWD $\overline{x}$. For each component k ($k \in [1, 3N_C]$) calculate the new corresponding component value of the mutated IWD $\hat{x}_k$ using the conditional equation :

   $$\hat{x}_k = \begin{cases} \overline{x}_k - s\left(\overline{x}_k - x^l\right) & t<r \\ \overline{x}_k - s\left(x^u - \overline{x}_k\right) & t \geq r \end{cases}$$

   where

   $$t = \frac{\overline{x}_k - x^l}{x^u - \overline{x}_k}$$

   Note that in each generation, all components of each IWD will have the same s and r and the value of each component, $\hat{x}_k$, in the mutated IWD will be calculated based on whether $t<r$ or $t \geq r$.

   (d) Relax $\hat{x}$ using L-BFGS.

   (e) Replace $\overline{x}$ if the energy of the relaxed $\hat{x}$ is more favourable.

2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Parameter Values : In this study, we use $p = 4$ while x^l and x^u are $-L/2$ and $L/2$, repectively, for each x, y, z component for a cubic bounding volume with side size L. For a spherical bounding volume, boundaries are defined using a polar coordinate system of a sphere centered in the origin with radius *rad*.

Laplace Crossover

The Laplace Crossover uses two randomly selected parent IWDs, $\overline{x1}$ and $\overline{x2}$, to generate two offspring IWDs, $\hat{x1}$ and $\hat{x2}$. The components of the offspring IWDs will be based on the distance between corresponding parent components and a random number satisfying a Laplace Distribution. Pseudo-code of Laplace Crossover is shown in Algorithm 2.

Algorithm 2 Pseudo-code of Laplace Crossover

1. Randomly select two IWDs $\overline{x1}$ and $\overline{x2}$ from the population.
   (a) For each component k ($k \in [1, 3N_C]$)
      i. Calculate random numbers $r, u \in U(1, 0)$
      ii. Calculate β_k satisfying the Laplace Distribution

$$\beta_k = \begin{cases} a - b \times log(u) & r \leq 0.50 \\ a + b \times log(u) & otherwise \end{cases}$$

      The parameters a and b are the location and scaling parameters, respectively.
      iii. Create two new IWDs, $\hat{x1}$ and $\hat{x2}$, by calculating their component values using :

$$\hat{x1}_k = \overline{x1}_k + \beta_k \left| \overline{x1}_k - \overline{x2}_k \right|$$

$$\hat{x2}_k = \overline{x2}_k + \beta_k \left| \overline{x1}_k - \overline{x2}_k \right|$$

   (b) Relax $\hat{x1}$ and $\hat{x2}$ using L-BFGS.
   (c) Sort relaxed offspring IWDs and parent IWDs according to energies.
   (d) Set $\overline{x1}$ and $\overline{x2}$ to be equal to the two IWDs with the lowest energies.
2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Parameter Values : In this study, the values of a and b are set to 1.0 and 0.50, respectively.

Inversion

Inversion is a very straightforward operator wherein a new IWD can be generated by selecting a portion of contiguous components from its string representation and inverting or flipping their positions. All IWDs in the population will undergo Inversion. Pseudo-code of Power Mutation is shown in Algorithm 3.

Algorithm 3 Pseudo-code of Inversion

1. For each IWD $\overline{x}$ in the population
   (a) Select random distinct component positions $c \in [1, 3N_C]$ and $d \in [1, 3N_C]$.
   (b) Set the larger value between c and d as $high$ and the smaller one as low.
   (c) From original IWD $\overline{x} = \{\overline{x}_1, \overline{x}_2, ..., \overline{x}_{low}, \overline{x}_{low+1}, ..., \overline{x}_{high-1}, \overline{x}_{high}, ..., \overline{x}_{3N_C}\}$, create new IWD $\hat{x}$ with :

$$\hat{x} = \{\overline{x}_1, \overline{x}_2, ..., \overline{x}_{high}, \overline{x}_{high-1}, ..., \overline{x}_{low+1}, \overline{x}_{low}, ..., \overline{x}_{3N_C}\}$$

   (d) Relax $\hat{x}$ using L-BFGS.
   (e) Replace $\overline{x}$ if the energy of the relaxed $\hat{x}$ is more favourable.
2. Go back to Step 1 until maximum number of generations, $maxGenerate$, is reached.

Geometric Mean

Geometric Mean generates a new IWD by taking a type of average of two parent IWDs. Specifically, average is taken by the root of the product of components of the parents. Pseudo-code of Geomertric Mean is shown in Algorithm 4.

Algorithm 4 Pseudo-code of Geometric Mean

1. Randomly select two IWDs $\overline{x1}$ and $\overline{x2}$ from the population.
   (a) Create a new IWD from $\overline{x1}$ and $\overline{x2}$ as :

$$\overline{x1x2} = \{abs(\overline{x1}_1\overline{x2}_1)^{1/2}, ..., abs(\overline{x1}_{3N_C}\overline{x2}_{3N_C})^{1/2}\}$$

   (b) Relax $\overline{x1x2}$ using L-BFGS.
   (c) Replace one of $\overline{x1}$ or $\overline{x2}$ if relaxed $\overline{x1x2}$ has more favourable energy.
2. Go back to Step 1 until maximum number of generations, $maxGenerate$, is reached.

Arithmetic Mean

Arithmetic Mean generates a new IWD by simply averaging the corresponding components of two randonly selected IWDs in the population. Pseudo-code of Arithmetic Mean is shown in Algorithm 5.

Algorithm 5 Pseudo-code of Arithmetic Mean

1. Randomly select two IWDs $\overline{x1}$ and $\overline{x2}$ from the population.
   (a) Create a new IWD from $\overline{x1}$ and $\overline{x2}$ as :
   $$\overline{x1x2} = \{0.50(\overline{x1}_1 + \overline{x2}_1), ..., 0.50(\overline{x1}_{3N_C} + \overline{x2}_{3N_C})\}$$
   (b) Relax $\overline{x1x2}$ using L-BFGS.
   (c) Replace one of $\overline{x1}$ or $\overline{x2}$ if relaxed $\overline{x1x2}$ has more favourable energy.
2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

2-Point Crossover

2-Point Crossover generates two new IWDs by exchanging contiguous portions of components from each parent IWD. Each new IWD will have two contigous portions from each of its parent IWD. Pseudo-code of 2-Point Crossover is shown in Algorithm 6.

Algorithm 6 Pseudo-code of 2-Point Crossover

1. Randomly select two IWDs $\overline{x1}$ and $\overline{x2}$ from the population.
   (a) Select a cutpoint component position c
   (b) Create two new IWDs $\hat{x1}$ and $\hat{x2}$ as :
   $$\hat{x1} = \{\overline{x1}_c, \overline{x1}_{c+1}, ..., \overline{x1}_{3N_C}, \overline{x2}_1, \overline{x2}_2, ..., \overline{x2}_{c-1})\}$$
   $$\hat{x2} = \{\overline{x2}_c, \overline{x2}_{c+1}, ..., \overline{x2}_{3N_C}, \overline{x1}_1, \overline{x1}_2, ..., \overline{x1}_{c-1})\}$$
   (c) Relax $\hat{x1}$ and $\hat{x2}$ using L-BFGS.
   (d) Sort relaxed offspring IWDs and parent IWDs according to energies.
   (e) Set $\overline{x1}$ and $\overline{x2}$ to be equal to the two IWDs with the lowest energies.
2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

N-Point Crossover

N-Point Crossover generates two new IWDs where each component of the IWD is chosen from either parent based on a random number. Pseudo-code of N-Point Crossover is shown in Algorithm 7.

Algorithm 7 Pseudo-code of N-Point Crossover

---

1. Randomly select two IWDs $\overline{x1}$ and $\overline{x2}$.
2. For each component k ($k \in [1, 3N_C]$) generated a random number $\delta \in U(0, 1)$
   (a) Create two new IWDs $\hat{x1}$ and $\hat{x2}$ as :

$$\hat{x1}_k, \hat{x2}_k = \begin{cases} \overline{x1}_k, \overline{x2}_k & \delta_k \leq 0.50 \\ \overline{x2}_k, \overline{x1}_k & \delta_k > 0.50 \end{cases}$$

   (b) Relax $\hat{x1}$ and $\hat{x2}$ using L-BFGS.
   (c) Sort relaxed offspring IWDs and parent IWDs according to energies.
   (d) Set $\overline{x1}$ and $\overline{x2}$ to be equal to the two IWDs with the lowest energies.
3. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Twinning

Twinning is a change on geometry of the cluster by a random rotation of all particles belonging on one of its hemisphere. All IWDs in the population will undergo Twinning. Pseudo-code of Twinning is shown in Algorithm 8.

Algorithm 8 Pseudo-code of Twinning

---

1. For each cluster $\overline{x}$ in the population
   (a) Set $\hat{x}$ to $\overline{x}$.
   (b) Randomly select a plane passing through the centre of mass of $\hat{x}$. This divides the cluster into two hemispheres.
   (c) Randomly select one side of the plane.
   (d) Randomly select an angle ϕ between 1 to 180.

(e) Rotate all the particles in $\hat{x}$ belonging within the hemisphere of the selected side by ϕ.

(f) Relax $\hat{x}$ using L-BFGS.

(g) Replace $\overline{x}$ if the energy of the relaxed $\hat{x}$ is more favourable.

2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Etch and Grow

Etch and Grow is a fine-tuning type of mutation where particles are removed from the cluster and gradually replaced by adding it back to random positions in the cluster. In Wolf, *et al* [Wolf & Landman, 1998], the particles are added back to the surface of the cluster. In MIWD, particles are placed in a random position in the interior of the cluster. The volume defining the interior of the cluster is bounded by the particles found on the surface. All IWDs in the population will undergo Etch and Grow. Pseudo-code of Etch and Grow is shown in Algorithm 9.

Algorithm 9 Pseudo-code of Etch and Grow

---

1. For each cluster $\overline{x}$ in the population

   (a) Set $\hat{x}$ to $\overline{x}$.

   (b) Remove m high-energy particles from $\hat{x}$ all at once.

   (c) For 1 to m

      i. Randomly place one particle in the interior of $\hat{x}$.

      ii. Relax the grown cluster using L-BFGS.

   (d) Set the etched and grown cluster to $\hat{x}$.

   (e) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.

2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Parameter Values : In this study, m is set to at least 30% of the size of the cluster being tested.

Grow and Etch

Grow and Etch works as the reverse of Etch and Grow. The cluster is first grown by adding particles on the cluster and slowly etching until the cluster is returned to its original number of particles. In Wolf, *et al* [Wolf & Landman, 1998], the particles are grown on the surface of the cluster. In MIWD, the particles are placed in the interior of the cluster. All IWDs in the population will undergo Grow and Etch. Pseudo-code of Grow and Etch is shown in Algorithm 10.

Algorithm 10 Pseudo-code of Grow and Etch

1. For each cluster $\overline{x}$ in the population
   (a) Set $\hat{x}$ to $\overline{x}$.
   (b) Randomly place m particles in the interior of $\hat{x}$ all at once.
   (c) For 1 to m
      i. Remove the highest-energy particle in $\hat{x}$.
      ii. Relax the etched cluster using L-BFGS.
   (d) Set the grown and etched cluster to $\hat{x}$.
   (e) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.
2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Parameter Values : In this study, m is set to at least 30% of the size of the cluster being tested.

2.7.2 Operators for Binary Lennard Jones Clusters

The Binary LJ system requires a specific set of perturbation operators to generate new clusters that would direct changes on its compositional aspect. Thus, the IWDs generated at the end of step 14 were tested on 4 different perturbation operations to find lower cluster energies for the Binary LJ system. One operator, called *Kneading* **(KNEAD)**, was taken from the study of Tao and co-workers [Tao *et al.*, 2011]. Another one, developed for this study **(CUTSPLICEVAR)**, is loosely based on the *Cut and Splice* genetic algorithm operator for structural optimization of Lennard Jones clusters in a paper by Deaven and Ho [Deaven & Ho, 1995]. The remaining two operators, **(EBS_HALB, EBS_HBLA)**, are energy-based swap operators which are also based on operators used in Tao and co-workers [Tao *et al.*, 2011]. An implementation of one of the above operators constitutes the MIWD+PerOp under the Binary LJ system. Details of these operators are presented in Algorithms 11 to 13.

Knead

Kneading is similar to Etch and Grow operator used in the LJ system except that all removed particles are added all at once instead of one at a time. All IWDs in the population will undergo Kneading. Pseudo-code of Kneading is shown in Algorithm 11.

Algorithm 11 Pseudo-code of Knead

---

1. For each cluster $\overline{x}$ in the population
   (a) Set $\hat{x}$ to $\overline{x}$.
   (b) Remove the top m highest-energy particles in $\hat{x}$.
   (c) Randomly place all removed particles in the interior of $\hat{x}$ at least a distance d away from the centre of the cluster.
   (d) Relax $\hat{x}$ using L-BFGS.
   (e) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.
2. Go back to Step 1 until maximum number of generations, $maxGenerate$, is reached.

Parameter Values : In this study, m was chosen to be 10% of the number of particles in the cluster. The value of d was chosen to be 2.0, a value similar to what was used in the study of Tao, *et al* [Tao *et al.*, 2011].

CutSpliceVar

Cut and Splice, as implemented in a GA-based method for the LJ system by Deaven and Ho [Deaven & Ho, 1995], starts by cutting a random plane from each of two parent clusters. The planes are not necessarily the same for both parents. The particles lying above the plane from one parent is then assembled with the particles below the plane from the other parent. If the cluster generated does not contain the correct number of particles, both planes are translated an equal distance in opposing directions normal to the cut plane. For the Binary LJ system, cutting and assembling parent segments this way is not easily adaptable as the probability of generating a cluster with different compositions is high. The operation must preserve the number of type A and type B particles in the child and parent clusters, thus a variant based on this operator was developed. In the variant, instead of a cutting plane, displaced type A (B) particles of one parent is assembled with displaced type B(A) particles of another parent. Since both parents have exactly the same number of type A and type B particles,

no adujustments are needed to be made in the composition. Pseudo-code of CutSpliceVar is shown in Algorithm 12.

Algorithm 12 Pseudo-code of CutSpliceVar

1. Randomly select two IWDs $\overline{x1}$ and $\overline{x2}$ from the population.
   (a) Using the expanded representation of $\overline{x1}$ and $\overline{x2}$, displace each component to move a distance up to $-\lambda$ or $+\lambda$ away from its current value.
   (b) Take all displaced type A (B) particles from $\overline{x1}$ and assemble it with all the displaced particles B (A) from $\hat{x1}$ and set it to IWD $\hat{x1}$ ($\hat{x2}$).
   (c) Relax $\hat{x1}$ and $\hat{x2}$ using L-BFGS.
   (d) Sort relaxed offspring IWDs and parent IWDs according to energies.
   (e) Set $\overline{x1}$ and $\overline{x2}$ to be equal to the two IWDs with the lowest energies.
2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Parameter Values : In this study, λ is set to $\sigma_{BB}(2^{1/6})(N^{1/3})$ where N is the number of particles in the cluster and σ_{BB} is the LJ parameter for particles of type B. The measure λ is adapated from the study of Marques, *etal* [Marques & Pereira, 2010] where it was used as a maximum range in which Cartesian coordinates of particle positions randomized.

EBS_HALB and EBS_HBLA

EBS_HALB and EBS_HALB are based on the *Smoothing* operator of Tao, *etal* [Tao *et al.*, 2011] and initially used in the Northby algorithm [Northby, 1987]. In the smoothing operator, $2N$ "vacant" sites are identified from the surface of a cluster using a probe atom which was chosen to be type B. A "vacant" site is defined as an unoccupied position on the surface of the cluster with the lowest energy contribution. The top highest-energy particles of the clusters are then moved one by one to one of the identified vacant sites. In MIWD, a simple variant was used where a high-energy particle of type A (B) is swapped with a low-energy particle of type B (A). All IWDs in the population will undergo EBS_HALB or EBS_HBLA.Pseudo-code of EBS_HALB (EBS_HALB) is shown in Algorithm 13.

Algorithm 13 Pseudo-code of EBS_HALB (EBS_HBLA)

1. For each cluster $\overline{x}$ in the population

(a) Set $\hat{x}$ to $\overline{x}$.

(b) Identify the highest-energy particle of type A (B) in $\hat{x}$.

(c) Identify the lowest-energy particle of type B (A) in $\hat{x}$.

(d) Swap the highest- (lowest-) energy particle of type A (B) with the lowest- (highest-) energy particle of type B (A).

(e) Relax $\hat{x}$ using L-BFGS.

(f) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.

2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

These perturbation operators were also tested in combinations, as illustrated in Figure 2.11 to identify the most effective perturbation strategy.

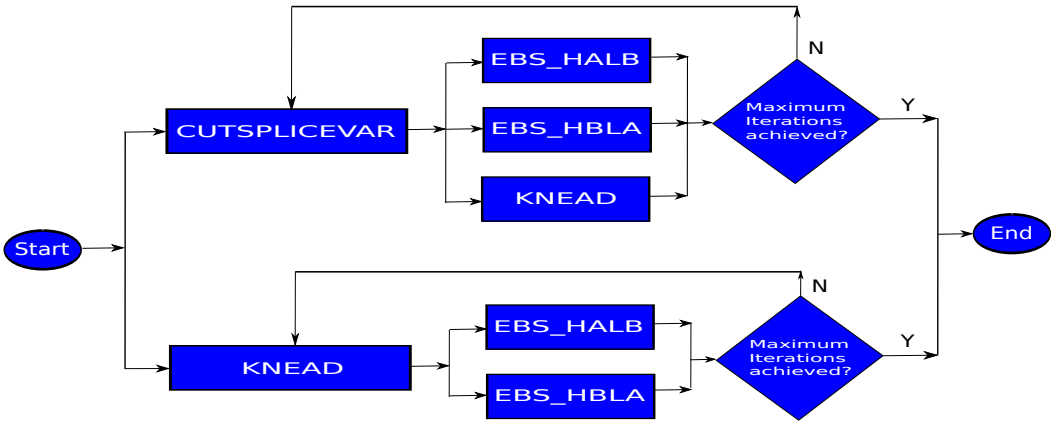


Figure 2.11: Flowchart of alternating application of perturbation operators used in the Binary Lennard Jones system.

2.7.3 Operators for Morse Clusters

Experiments on Morse clusters only used the Grow and Etch [Wolf & Landman, 1998] perturbation operator for all cluster sizes tested. Under the Morse system, only MIWD+PerOp was implemented.

2.7.4 Operators for Janus Clusters

Four different types of perturbation operators were used in Phase 2 for the Janus system : *Orientation Mutation*, *Displacement Mutation*, *Neighbour Optimization*, and *Grow Etch Mutation*. Orientation and Displacement mutations were also used in the cluster study of Janus fluids by Fantoni and co-workers [Fantoni *et al.*, 2014] and in investigation of one-patch colloidal particles by Sciortino

and co-workers [Sciortino *et al.*, 2010]. Neighbourhood optimization, based in Orientation Mutation, is believed to be applied first in this study. Details of these operators are presented in Algorithms 14 to 17.

Orientation Mutation

Orientation Mutation is a simple random change on the orientation of all particles in a Janus clusters. All IWDs in the population will undergo Orientation Mutation. Pseudo-code of Orientation Mutation is shown in Algorithm 14.

Algorithm 14 Pseudo-code of Orientation Mutation

1. For each cluster $\overline{x}$ in the population.
   (a) Set $\hat{x}$ to $\overline{x}$.
   (b) For each particle in cluster $\hat{x}$
      i. Generate a new uniformly distributed random particle orientation.
   (c) Relax $\hat{x}$ using L-BFGS.
   (d) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.
2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Displacement Mutation

Displacement Mutation is a simple random change on the position of all particles while fixing their orientation. The random change in the position is similar to the displacement strategy used in CutSpliceVar. All IWDs in the population will undergo Displacement Mutation. Pseudo-code of Displacement Mutation is shown in Algorithm 15.

Algorithm 15 Pseudo-code of Displacement Mutation

1. For each cluster $\overline{x}$ in the population.
   (a) Set $\hat{x}$ to $\overline{x}$.
   (b) For each particle in cluster $\hat{x}$
      i. Using the expanded representation of $\hat{x}$, displace each component to move a distance up to $-\lambda$ or $+\lambda$ away from its current value.
   (c) Relax $\hat{x}$ using L-BFGS.

(d) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.

2. Go back to Step 1 until maximum number of generations, $maxGenerate$, is reached.

Parameter Values : In this study, λ is set to $\sigma_{BB}(2^{1/6})(N^{1/3})$ where N is the number of particles in the cluster and σ_{BB} is the LJ parameter for particles of type B. The measure λ is adapated from the study of Marques, *etal* [Marques & Pereira, 2010] where it was used as a maximum range in which Cartesian coordinates of particle positions randomized.

Neighbourhood Mutation

Neighbourhood Mutation optimizes a particle's orientation on a randomly chosen particle around its neighbourhood. Only a fraction of the cluster's particles are orientationally optimized. All IWDs in the population will undergo Neighbourhood Mutation. Pseudo-code of Displacement Mutation is shown in Algorithm 16.

Algorithm 16 Pseudo-code of Neighbourhood Mutation

---

1. For each cluster $\overline{x}$ in the population.

   (a) Set $\hat{x}$ to $\overline{x}$.

   (b) Set $numToOptimize = floor(N/2)$

   (c) For 1 to $numToOptimize$

      i. Randomly choose a particle p_{opt} from $\hat{x}$ that has not been previously optimized yet.

      ii. Select the closest neighbour p_{neigh} of the p_{opt} that has not been previously optimized yet.

      iii. For 1 to $max\_iterations$

         A. Generate a new uniformly distributed random particle orientation for particle p_{opt}.

         B. If the orientation measure, $MVang$, of particle p_{opt} to particle p_{neigh} is greater than the previous orientation measure, set the particle orientation of p_{opt} to the new one.

   (d) Relax $\hat{x}$ using L-BFGS.

   (e) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.

2. Go back to Step 1 until maximum number of generations, $maxGenerate$, is reached.

Parameter Values : In this study, the number of particles *numToOptimize* that were orientationally optimized to their neighbours were half the size of the cluster. The function $floor(expression)$ returns the largest integer value less than or equal to *expression*. *max_iterations* were set to at least 5000.

Grow and Etch for Janus Clusters

This operator is based on the same operator described by Wolf and Landman [Wolf & Landman, 1998] and used as a perturbation operator for LJ and Morse clusters in this study. For the Janus system, in addition to randomly placing particles during the growing stage, a random orientation is also generated for each particle. All IWDs in the population will undergo Grow and Etch Mutation. Pseudo-code of Displacement Mutation is shown in Algorithm 17.

Algorithm 17 Pseudo-code of Grow and Etch for Janus clusters

1. For each cluster $\overline{x}$ in the population.
   (a) Set $\hat{x}$ to $\overline{x}$.
   (b) Randomly place m Janus particles in the interior of $\hat{x}$.
   (c) For 1 to m
      i. Remove the highest-energy particle in $\hat{x}$.
      ii. Relax the etched cluster using L-BFGS.
   (d) Replace $\overline{x}$ if the energy of $\hat{x}$ is more favourable.
2. Go back to Step 1 until maximum number of generations, *maxGenerate*, is reached.

Parameter Values : In this study, m is set to at least 30% of the size of the cluster being tested.

A combination of perturbation operators (CombiOp) were used to further find lower energies of clusters generated by MIWD. Figure 2.12 shows the combinations used in this study. We shall refer to the combination Orientation Mutation and Displacement Mutation as *OrDis*, Neighbour Optimization and Displacement Mutation as *NeighDis*, Grow Etch and Neighbour Optimization as *GrowEtchNeigh*, Grow Etch and Orientation Optimization as *GrowEtchOr*, Neighbour Optimization and Grow Etch as *NeighGrowEtch*, and Orientation Optimization and Grow Etch as *OrGrowEtch*.

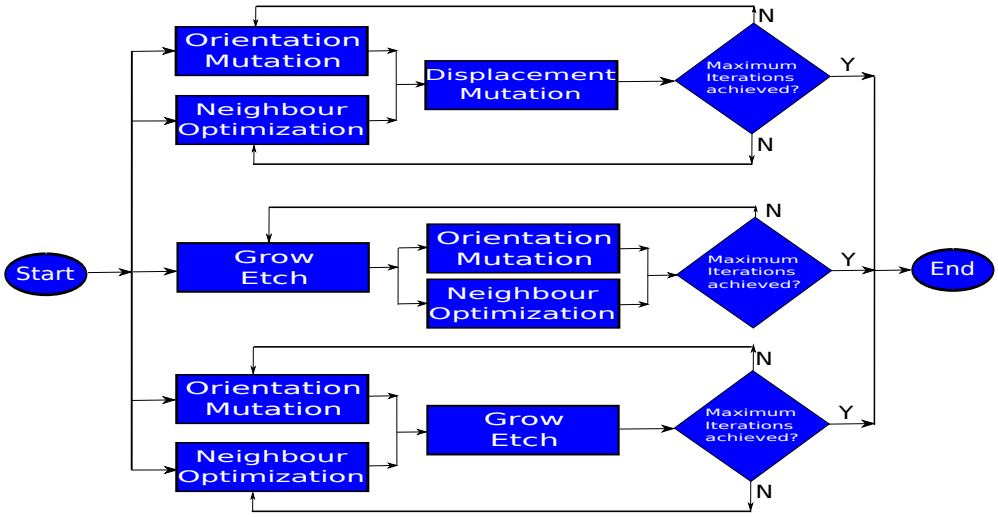


Figure 2.12: Flowchart of alternating application of perturbation operators used in the Janus system.

2.8 Implementation Details

Experiments presented in this paper were carried out using the Centre for Scientific Computing High Performance Computing Facilities at the University of Warwick. These facilities were provided by the MidPlus Regional Centre of Excellence for Computational Science, Engineering and Mathemtatics, under EPSRC grant EP/K000128/1.

The algorithm was coded in C and Fortran using the GNU Compiler Collection version 4.3.4. Fortran Character String utilities of G. Benthien [Benthien, n.d.] obtained online `http://gbenthien.net/strings/` were used in file reading. The C library for L-BFGS was obtained from `http://www.chokkan.org/software/liblbfgs/` [Okazaki, n.d.]. The Fortran version of L-BFGS used for this study was obtained from `http://users.iems.northwestern.edu/~nocedal/lbfgs.html` [Nocedal, n.d.]. The random number generator Fortran code was obtained from `http://web.ph.surrey.ac.uk/fortweb/glossary/random_seed.html` (Appendix B).

The following graphing utilities were used : R v.3.0.0 (2013-04-03) R Core Team [2013], Gnuplot version 4.4 [Williams & Kelley, 2010], Matlab 8.1.0.604 (R2013a) [MATLAB, 2013], Inkscape 0.48.2 r9819 [Bah *et al.*, n.d.] and Visual Molecular Dynamics for LINUXAMD64 Version 1.9.1 [Humphrey *et al.*, 1996]. Thesis was typeset using LaTex and pdf file was generated using pdfTex 3.14159265-2.6-1.40.16 [Lamport, n.d.]. Text editor used in writing codes was GNU Emacs 22.3.1 [Stallman & Steele, 1985].

Chapter 3

Modified Intelligent Water Drops (MIWD) and Bounding Volumes

3.1 Introduction

The mathematically simple, yet NP Hard-categorized, model of the Lennard-Jones potential makes it a challenging optimization problem for finding globally optimal solutions. It has been used as a benchmark for robustness of new global optimization methods. More importantly, if the new global optimizer could find the notoriously difficult clusters namely, LJ_{38}, LJ_{75-77}, LJ_{98} and $LJ_{102-104}$, this optimizer could be a promising method for finding optimal configurations for larger cluster sizes. This being said, LJ clusters will be used as a testbed for the effectivity of MIWD with Perturbation Operators (MIWD+PerOp) in finding optimal configurations which corresponds to lowest energies of atomic clusters.

The following subsections will include test runs showing the ability of Phase 1 of MIWD+PerOp to find lower energies as iterations progress. Furthermore, test runs on the different bounding volumes and perturbation operators for specific LJ cluster sizes will be presented in this chapter as well. The relaxation method used in this chapter is the quasi-Newton Limited Memory BFGS (L-BFGS). Initially, relaxation was done using quasi-Newton limited memory conjugate gradient method from NAG C Library but comparison of tests with L-BFGS showed that the latter generated desirable (lower) energies in a shorter period of time. A chapter showing the results of this comparison can be found following this chapter. In the succeeding chapters, best results of tests in this chapter will be used to run tests on different perturbation operators (Phase 2 of MIWD+PerOp). The best combination of bounding volume and perturbation operator, resulting

from test runs, will then be used as final settings of MIWD+PerOp for LJ clusters up to size 104 in another chapter.

3.2 MIWD Phase 1 and Bounding Volumes

The ability of the first phase of MIWD to locate lower energies on repeated iterations of MIWD Phase 1 is shown here. These tests also show the performances of the different bounding volumes as applied on LJ_{13}, the first complete icosahedron in the LJ system and two relatively difficult cluster sizes LJ_{38}, truncated octahedron, and LJ_{98}, a tetrahedron.

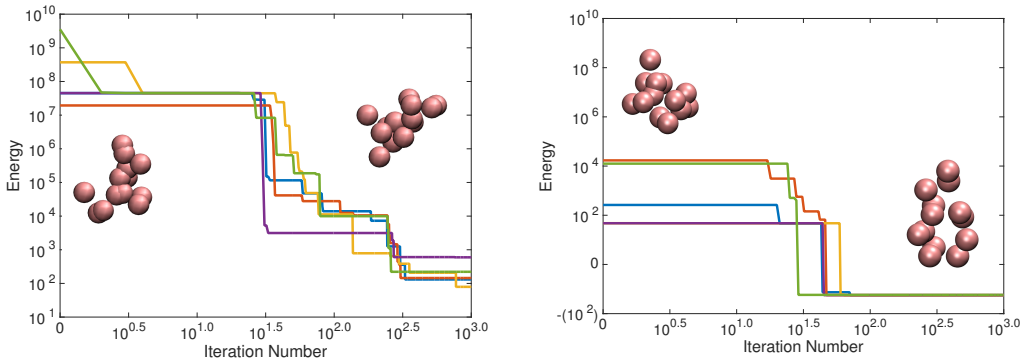
Five runs with 650 initial particle positions were used for each type of bounding volume (Cai, Chen, Hod and Wales). It is to be noted that none of the clusters have been relaxed or subjected to any relaxation method at the end of every iteration. Table 3.1 shows the number of iterations made for each cluster size tested.

Table 3.1: Number of iterations used for the different bounding volumes for MIWD Phase 1.

Cluster Size	Iterations
13	1000
38	{2000,5000,10000}
98	10000

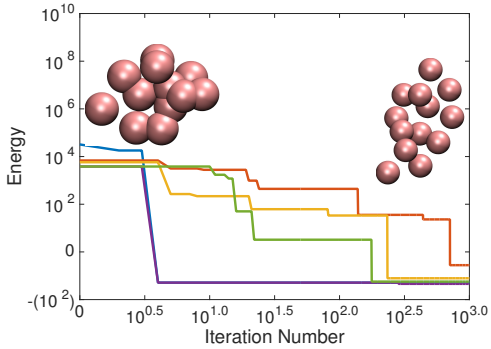
3.2.1 LJ_{13} for 1000 iterations

Figure 3.1 shows the resulting energies as iterations progress for 4 different bounding volumes as tested under LJ_{13}. It can be observed that MIWD is able to significantly move from a higher potential energy to a lower potential energy. In most figures, decline of energy starts to speed up after the 30th iteration. The plateau at the first 100 iterations is largely due to the fact that the algorithm is still building information about the strengths of the connectivities ($soil(i, j)$) of atom sites. It is safe to say that for this instance, after 30 iterations, MIWD has been able to determine which connectivities are more likely generating the lower, if not the lowest, energies. In the same figure, starting and final configurations of one of the runs are overlaid. The configurations on the left of each subplot are the starting configurations while the configurations on the right are the final configurations. Starting and final configurations show the ability of MIWD to find a more desirable distance of separation between particles from initially overlapping particle positions as iterations progress (i.e. particle repulsion is illustrated as

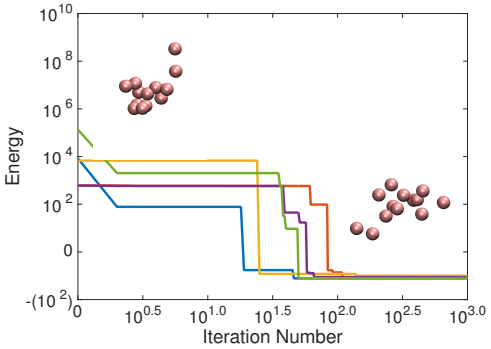


(a) Cai bounding volume

(b) Chen bounding volume



(c) Hod bounding volume



(d) Wales bounding volume

Figure 3.1: Five runs (1000 iterations each) of MIWD Phase 1 showing performances of 4 bounding volume as tested on LJ_{13}. Initial (left of each subplot) and final (right of each subplot) configurations are overlayed as well.

iterations progress). This is clearly illustrated in subfigures 3.1a and 3.1b.

Inspecting at the individual bounding volume characteristics, for size 13, the Cai bounding volume would allow the initial particles to be scattered on a sphere with an approximate radius of 1.80 units (see Equation 2.24). In Wales bounding volume, particle positions are scattered on a sphere with a fixed radius of 5.5 units, while in Chen and Hod, particle positions are contained within a cube with side lengths 4 and 3 units, respectively. This means that in the Cai bounding volume, particle positions are more likely positioned closer to each other compared to the other bounding volumes. In fact, the initial particle positions in Cai bounding volumes are always more compact than the Wales bounding volumes, at least up to all the LJ cluster sizes tested in this study. On the other hand, the Cai bounding volume is more compact than the Chen and Hod bounding volumes at LJ cluster size less than 15 and less than 47, respectively. The compactness of the volume at which the initial particle sites were scattered under the Cai bounding volume for size 13 affected the energies of the generated clusters. Looking closely at Figure 3.1, Cai bounding volume (Figure 3.1a) generated, on the average, higher starting and final energies. There is a jump of around 8 orders

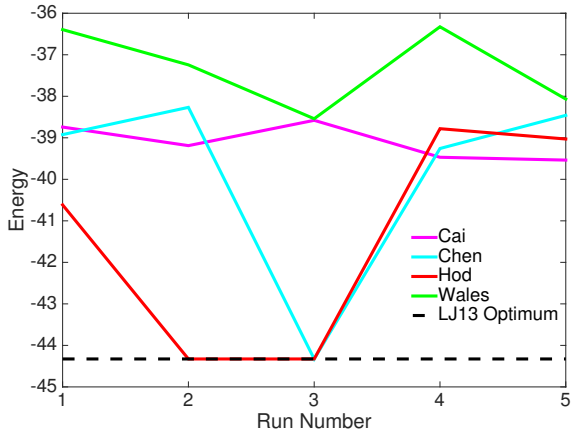


Figure 3.2: Relaxed configurations after the 1000th iteration for the LJ_{13} test runs. Wales bounding volume appears to generally have higher energies for its relaxed configurations. Only Chen and Hod bounding volumes were able to hit the global optima.

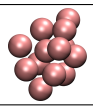
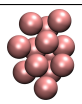
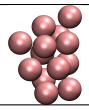
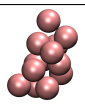
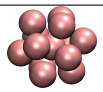
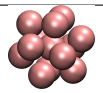
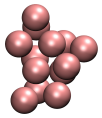
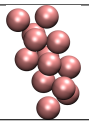
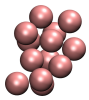
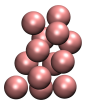
of magnitude between the initial energies and final energies under the Cai bounding volume while there is only about 6 orders of magnitude for Chen, Hod and Wales bounding volumes (Figures 3.1b, 3.1c and 3.1d, respectively). Chen, Hod and Wales bounding volumes started at relatively lower initial energies, ranging between 10 and 10^5 energy units, than Cai.

Furthermore, the trend in energy decrease under the Cai bounding volume is more gradual. Chen and Wales bounding volumes are showing sharp plunges after the 30th iteration and seems to plateau until the 1000th iteration while Cai bounding volume shows having to visit more high energy configurations after the 30th and before starting to plateau from the 300th. The final energies obtained using the Cai bounding volume in any of the 5 runs, however, failed to go even below zero energy units while the rest of the bounding volumes stopped at negative energies albeit not the global optimal energies. It was found out, however, that relaxing the final structures (structures at the 1000th iteration) of Chen and Hod bounding volumes lead to some runs hitting the global optimum with energy -44.326801 energy units (Figure 3.2 and Table 3.2). For the rest of the bounding volumes, their relaxed final structures obtained suboptimal energies.

There is no discernable difference between Chen, Hod and Wales bounding volume both in terms of iteration performance and starting energy configurations.

The observations above tell us of the ability of MIWD Phase 1 to move from a high energy configuration to a lower one. Furthermore, it is worth noting that the Che, Hod and Wales bounding volumes provide us with a good enough bounding volume for which starting energies are kept to a minimum and final energies are closer enough to the global optima when relaxed.

Table 3.2: Configurations corresponding to the relaxed geometries at the end of each run for LJ_{13} in the 1000 iterations case.

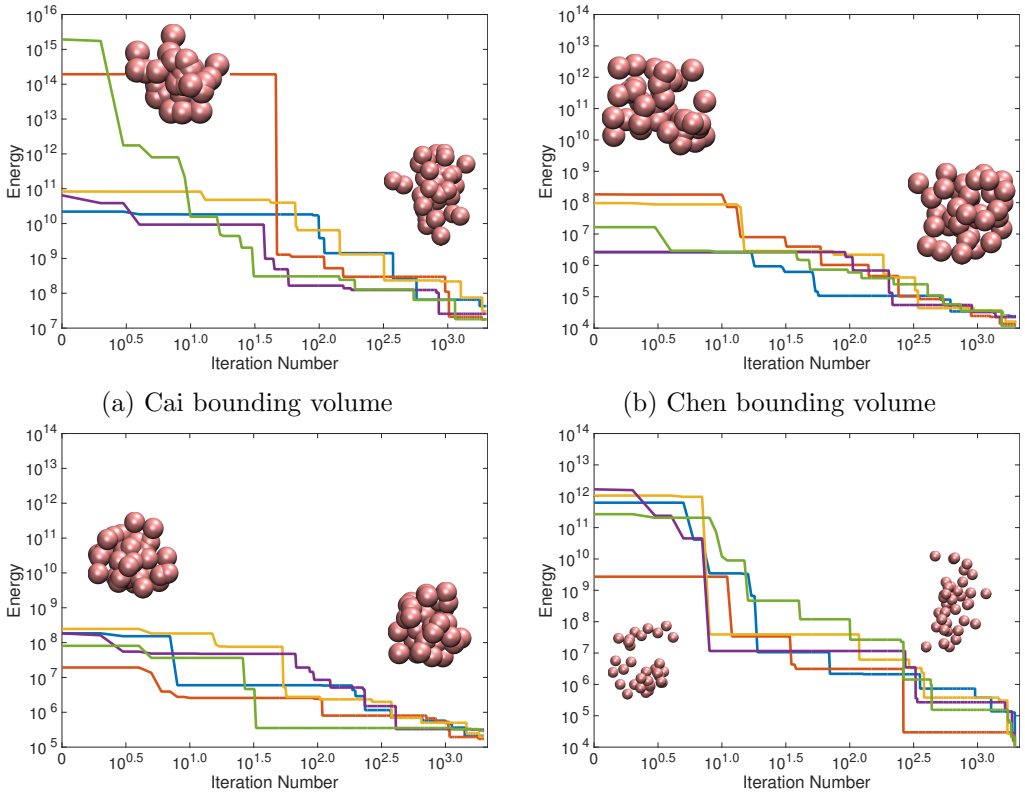
Bounding Volume	Run 1	Run 2	Run 3	Run 4	Run 5
Cai					
Chen			 Optimal		
Hod		 Optimal	 Optimal		
Wales					
LJ_{13} Global Optimum					

3.2.2 LJ_{38} for 2000, 5000 and 10000 iterations

Tests were also done on LJ_{38} to determine how well MIWD Phase 1 performs on a special case of a truncated octahedron geometry as opposed to the icosahedral geometry of LJ_{13}. Results for an initial test of 5 runs with 2000 iterations each are shown in Figure 3.3. It is clear from the subfigures that MIWD Phase 1 has the ability to proceed from high to lower energy configurations as iterations progress.

Putting all the succeeding observations in perspective, for LJ size 38, the radius of the sphere at which initial particle sites were scattered under the Cai bounding volume was approximately 2.36 units (see Equation 2.24), fixed radius of 5.5 units for the Wales bounding volume, and side lengths of 4 and 3 units for the cubic bounding volumes Chen and Hod, respectively. Given these measures, at similar number of initial particle sites, it can be expected that the Cai bounding volume will have more particles situated closer to each other, thus generating more highly energetic interactions, compared to the Wales and Hod bounding volumes. The Cai spherical bounding volume, however, has a larger volume than the Chen cubic bounding volume for LJ size 38. Nevertheless, the difference between initial and final energies for the Cai bounding volume is largest at about 8 orders of magnitude while only about 3-4 orders of magnitude for Chen, Hod and Wales except for the latter where one run jumped from an initial energy of $10^{11.5}$ to 10^{4} (Figure 3.3d).

Unlike the LJ_{13} test runs, there is no clear indication of a point in the



(a) Cai bounding volume

(b) Chen bounding volume

(c) Hod bounding volume

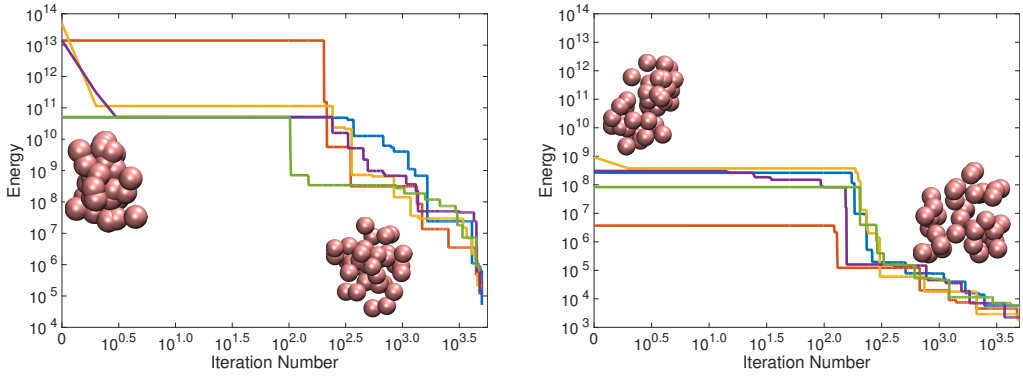
(d) Wales bounding volume

Figure 3.3: Five runs (2000 iterations) of MIWD Phase 1 showing performances of 4 bounding volumes as tested on LJ_{38}.

iterations where energy drop started in all bounding volumes. However, a good look at the downward trend of all energies in all bounding volumes show that furthering beyond 2000 iterations may provide us with lower energies. Thus, further tests were done with iterations increased to 5000 and 10000.

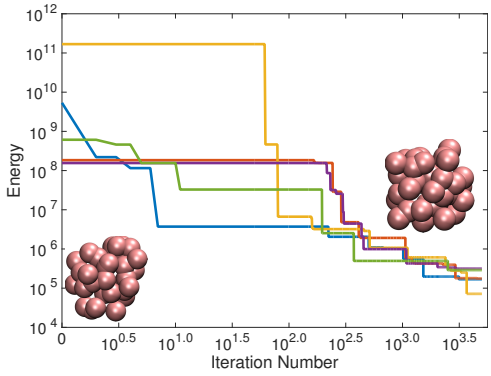
Figures 3.4 and 3.5 show the test runs for LJ_{38} 5000 and 10000 iterations, respectively. It is to be noted that the iterations in these plots are independent from the runs in Figure 3.3 (i.e. runs in Figures 3.4 and 3.5 have been restarted from iteration 1). It is evident from Figure 3.4 that lower energies may possibly still be discovered after the 5000th iterations. In all bounding volumes, final energies start to plateau after around the 5500th iteration (see Figure 3.5) with Wales as the only type of bounding volume to go below the negative energy mark in all its runs.

Relaxing the final energies of all the runs in all bounding volumes, it was observed that Wales' relaxed configurations, on the average, generated higher energies in all three cases (2000, 5000 and 10000 iterations). This is shown in Figure 3.6 where the putative global optimum energy line for LJ_{38} is plotted along with the relaxed configuration energies for all bounding volumes.

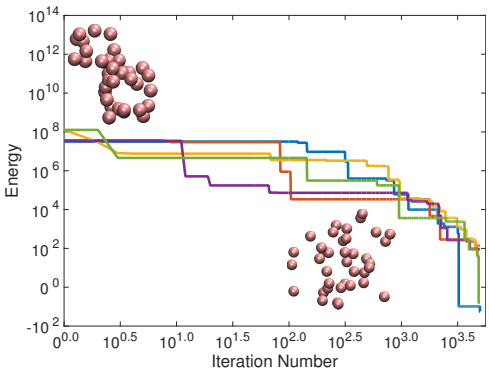


(a) Cai bounding volume

(b) Chen bounding volume

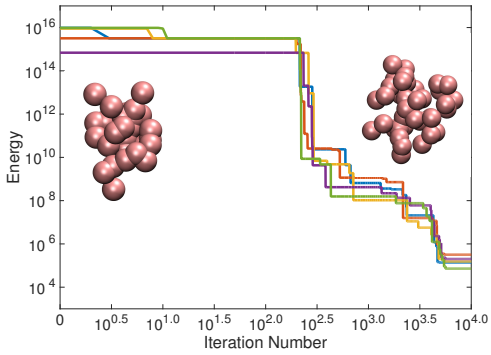


(c) Hod bounding volume

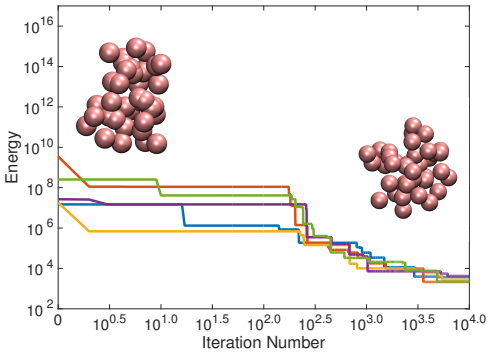


(d) Wales bounding volume

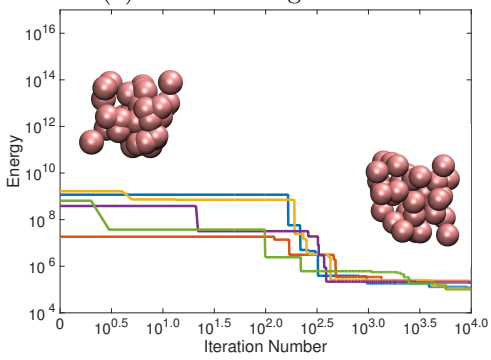
Figure 3.4: Five runs (5000 iterations) of MIWD Phase 1 showing performances of 4 bounding volumes as tested on LJ_{38}.



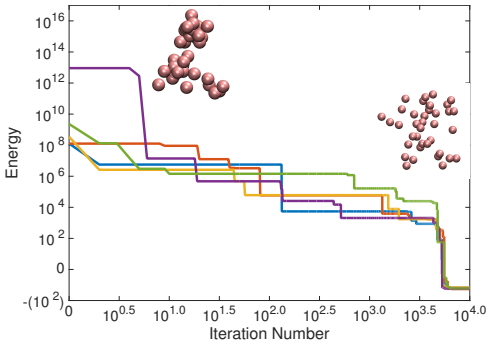
(a) Cai bounding volume



(b) Chen bounding volume

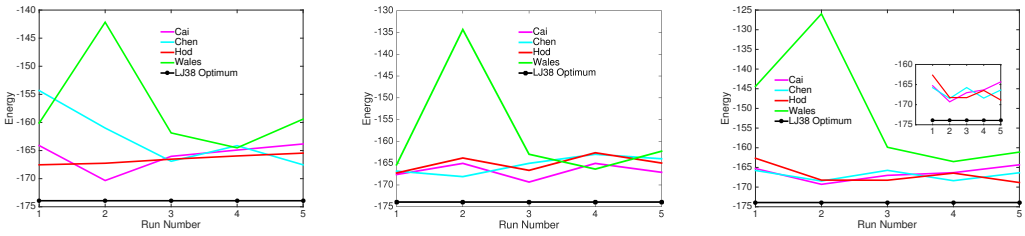


(c) Hod bounding volume



(d) Wales bounding volume

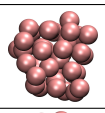
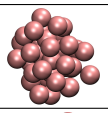
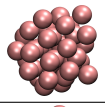
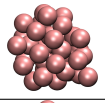
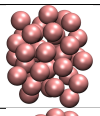
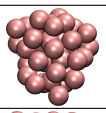
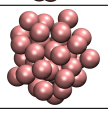
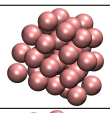
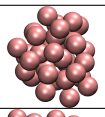
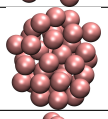
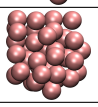
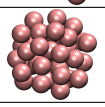
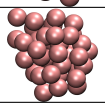
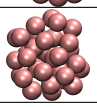
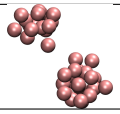
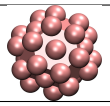
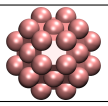
Figure 3.5: Five runs (10000 iterations) of MIWD Phase 1 showing performances of 4 bounding volumes as tested on LJ_{38}.



(a) (b) (c)

Figure 3.6: Relaxed configurations after the (a) 2000th, (b) 5000th, (c) 10000th iterations for the LJ_{38} test runs. Wales bounding volume appears to generally have higher relaxed energies for its relaxed configurations on the 5000th and 10000th iterations while significantly lower in energies in the test runs with the lowest (2000) iterations.Closer comparison of Cai, Chen and Hodgson energies shows no discernible difference on relaxed energies in all runs.

Table 3.3: Configurations corresponding to the relaxed geometries at the end of each run for LJ_{38} in the 10000 iterations case.

Bounding Volume	Run 1	Run 2	Run 3	Run 4	Run 5
Cai					
Chen					
Hod					
Wales					
LJ_{38} Global Optimum					

Across all bounding volumes, only test runs involving 10000 iterations (Figure 3.5) provided a more desirable separation of particles at the end of each run. For the 2000 and 5000 iterations, final configurations show a large percentage of overlapping particles with the exception of Wales bounding volume. These overlaps are clearly visible in subfigures 3.3c, 3.4b and 3.4c.

Despite generating lower energies after 10000 iterations, final energies of unrelaxed configurations were still high enough. Relaxed structures of the final run configurations, although considerably lower in energy than the final unrelaxed configurations, did not hit the putative global optima. This observation, both for the larger test cases LJ_{38} and LJ_{98}, necessitates an additional process to further explore the potential energy surface. This time, however, adding extra iterations of Phase 1 does not appear to be a good option as iterations can plateau at some point, instead an extra step (Phase 2) to locally explore the final configurations was implemented.

3.2.3 LJ_{98} for 10000 iterations

Phase 1 of MIWD was tested in an even larger cluster and a known difficult cluster due to its tetrahedral geometry, the LJ_{98}. In this test, the runs were iterated 10000 times. Decline of energy as iterations progress is evident in all bounding volumes although final energies are undesirably high and particle separation for final structures are far from optimal. There are obvious overlapping of particles on the final structures in Figures 3.7a, 3.7b and 3.7c.

The radius of the sphere under the Cai bounding volume at LJ size 98 is approximately 3.04 units (see Equation 2.24). This makes the Cai bounding

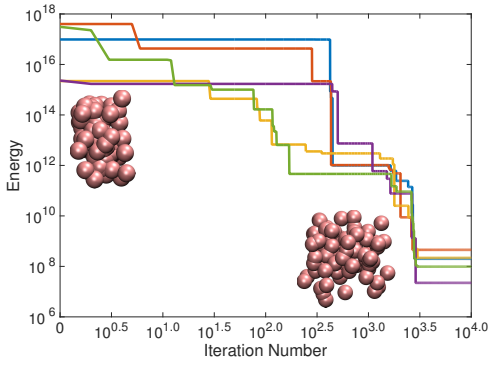
volume larger than the Chen and Hod cubic bounding volumes (with side lenghts of 3 and 4 units, respectively) at LJ cluster size 98. The spherical Wales bounding volume was still set at a fixed radius of 5.5 units making it a larger volume than the Cai bounding volume. Despite this change in volume size, Cai bounding volume still generated the largest energy difference, on the average, between the initial and final energies with about 10 orders of magnitude difference. On the other hand, the difference between initial and final energies for Chen and Hod bounding volumes are between 2-8 orders of magnitude while Wales is between 7-10 orders of magnitude. Although these differences show the ability of the algorithm to find lower energies despite starting high-energy configurations, the Cai bounding volume is the least desirable among the 4 bounding volumes as the high-energy starting configurations would require the algorithm to work more on finding better connectivities. Chen, Hod and Wales bounding volumes provided sufficient volumes for which starting configurations are sufficiently high in energy in the potential energy surface to be able to navigate to lower energy configurations but not high enough that would require the algorithm to work more.

Furthermore, as observed in previous test cases, the Wales bounding volume generated the average lowest unrelaxed energies while generating the highest relaxed energies (Figure 3.8 and Table 3.4).

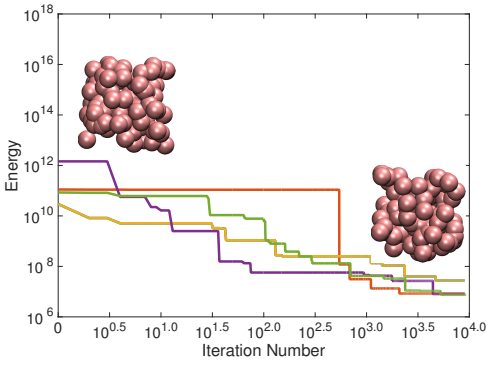
3.3 Summary and Conclusion

Phase 1 of MIWD have been tested on 3 cluster sizes, namely LJ_{13}, LJ_{38}, and LJ_{98}. The two latter sizes were chosen as test cases particularly because they are known to be difficult clusters which do not conform to the usual icosahedral theme of majority of optimal geometries for LJ clusters. Performance of different bounding volumes were also tested across all test cases.

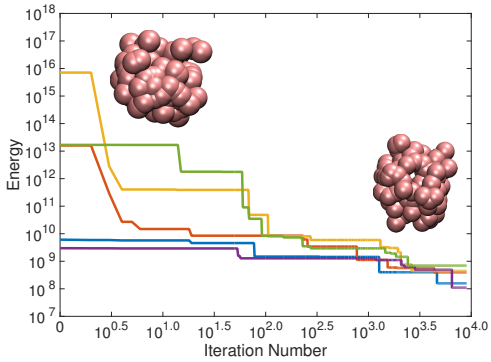
Results of experiments showed that MIWD was able to generate lower energies as iterations progress for all test cases and across all bounding volumes. Comparisons of bounding volumes shows that Cai bounding volume, on the average, generates clusters with high energies at the start and end of runs. Configurations at the end of the runs were also relaxed and results showed there was no discernible difference among Chen, Hod and Cai bounding volumes while Wales generated configurations with higher energies. For the LJ_{13} case, relaxed configurations of some runs in Hod and Chen bounding volumes were able to hit the global optimum. None of the relaxed configurations in any of the bounding volumes were able to hit the global optima for cases LJ_{38} and LJ_{98}. Furthermore, as expected of an iterative and stochastic algorithm, as the dimensionality of the



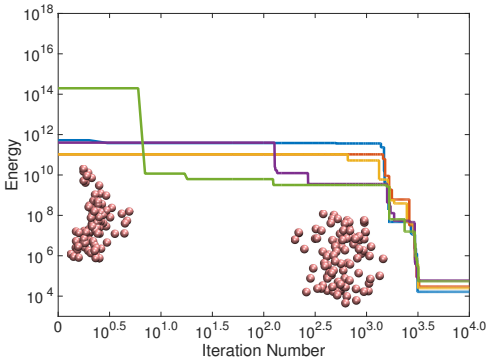
(a) Cai bounding volume



(b) Chen bounding volume



(c) Hod bounding volume



(d) Wales bounding volume

Figure 3.7: Five runs (10000 iterations) of MIWD Phase 1 showing performances of 4 bounding volumes as tested on LJ_{98}.

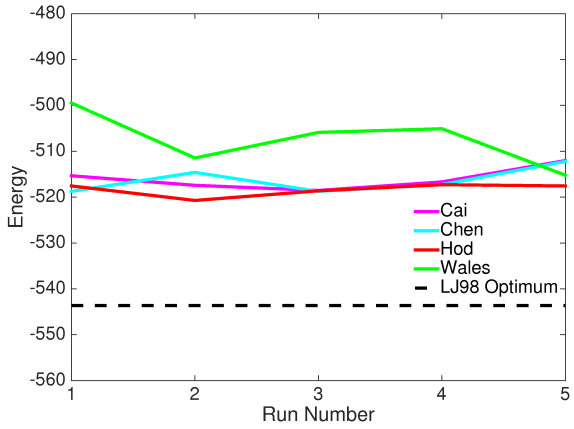
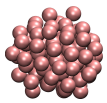
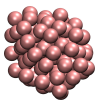
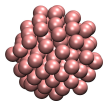
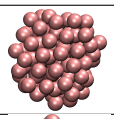
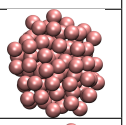
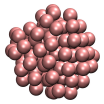
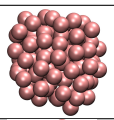
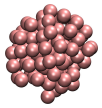
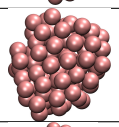
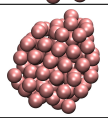
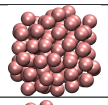
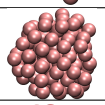
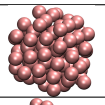
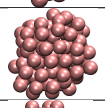
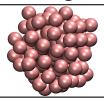
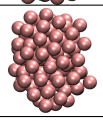
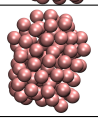
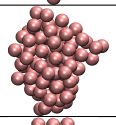
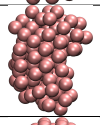
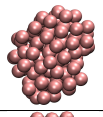
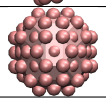
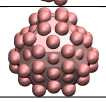
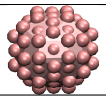


Figure 3.8: Relaxed configurations after the 10000th iteration for the LJ_{98} test runs. Wales bounding volume appears to generally have higher energies for its relaxed configurations. None of the bounding volumes generated relaxed configurations that hit the global optimum.

test problem increases, more iterations are also required.

Table 3.4: Configurations corresponding to the relaxed geometries at the end of each run for LJ_{98} in the 10000 iterations case.

Bounding Volume	Run 1	Run 2	Run 3	Run 4	Run 5
Cai					
Chen					
Hod					
Wales					
LJ_{98} Global Optimum					

Chapter 4

Relaxation Methods

4.1 NAG Conjugate Gradient Method (NAG CGM) and Limited Memory BFGS (L-BFGS)

An effective relaxation method that minimizes an unconstrained nonlinear function of several variables is an important feature of stochastic algorithms for atomic cluster optimization. When the solution space is as complex as the potential energy surface, relaxation methods allow solutions returned by exploratory stochastic algorithms to lie on a nearby, possibly lowest lying, basin. The two most used relaxation methods in a number of atomic cluster optimization studies are conjugate gradient method (CGM) [Deaven & Ho, 1995;Deaven *et al.*, 1996;Wales & Doye, 1997;Wolf & Landman, 1998;Locatelli & Schoen, 2001;Leary, 2000;Iwamatsu & Okabe, 2004], or its variants, and L-BFGS [Liu & Nocedal, 1989;Maier *et al.*, 1992;Hartke, 1999;Cai & Shao, 2002;Cai *et al.*, 2002a;Lee *et al.*, 2003;Xiang *et al.*, 2004;Zhou *et al.*, 2005;Chen *et al.*, 2010]. First derivatives, or an "acceptable" finite difference approximation, are required to use the CGM or the L-BFGS. They are intended for use on large scale problems which is an inherent characteristic of the atomic cluster optimization problem.

The NAG Conjugate Gradient Method (NAG CGM) (Appendix C) and Limited Memory - BFGS (L-BFGS) (Appendix A) are algorithms for unconstrained optimization. Both algorithms need the objective function and its gradient to work. When no analytic gradient can be provided, numerical differentiation can be used instead. NAG CGM and L-BFGS are two of the most used relaxation methods in the studies mentioned in this thesis (Section 1.5.2).

The L-BFGS algorithm builds a sufficiently good quadratic model of the objective function. It stores the last M function value/gradient pairs to build positive definite Hessian approximation which is then used to make the quasi-Newton step. If there is no sufficient decrease of the function value/gradient, the

search is done along a line in the direction of the step. The positive definiteness of the Hessian approximation allows for efficient calculations. Regardless of the function curvature, the algorithm always generates a symmetric positive definite matrix and the quasi-Newton direction will always be a descent step. The CGM, on the other hand, does not build a quadratic model of the objective function. In CGM, the function is being optimized along a line and the direction of search is a linear combination of the current gradient vector and the previous search direction.

Optimization of computationally cheap functions is more desirable using CGM than in L-BFGS due to the large storage overhead requirement in the latter. L-BFGS, on the other hand, requires less function evaluations in one iteration than CGM and thus a more desirable choice for computationally expensive optimization problems. At extremely ill-conditioned problems, convergence speed of CGM is only decreased while L-BFGS degenerates to steepest descent method. In this case, a good preconditioner is needed. Changes in the preconditioner, however, have different effects on both CGM and L-BFGS. In the former, all information about the function curvature accumulated so far is lost while in L-BFGS, all information are retained regardless of the change in the preconditioner.

Initial tests of MIWD+PerOp used the NAG CGM (Appendix C) as a relaxation method but results were not desirable on those runs and errors were consistently produced. As the second most used relaxation methods in literature for atomic cluster optimization, L-BFGS (Appendix A) was considered next. Although MIWD+PerOp and MIWD+CombiOp used L-BFGS in its final runs, we compare the performances of NAG CGM and L-BFGS on selected test cases and show what led to the selection of the latter as the final relaxation method in this study.

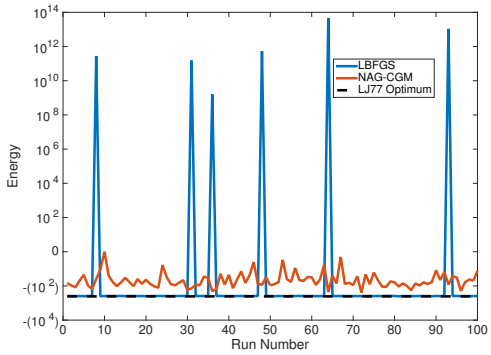
NAG CGM and L-BFGS were run on the same set of randomly generated atom sites for 100 independent runs and tested on different LJ cluster sizes, namely, LJ_{77}, LJ_{98}, LJ_{102}, LJ_{103}, and LJ_{104}. These cluster sizes were chosen due to the fact that they are particularly difficult non-icosahedral LJ clusters. After generation of atom sites, the configurations were immediately subjected to the relaxation methods. No other movement of sites were done after relaxation, thus obtaining the global optimum is not expected nor is the focus of this test. To avoid having to test all possible combinations of bounding volumes and test clusters, this experiment chose one representative from each category of bounding volumes (cubic and spherical). Based on the results from chapter 3, Wales is chosen as the spherical bounding volume over Cai as this resulted into lower energies both in the start and final iterations for unrelaxed configurations. There was no discernible difference between Hod and Chen cubic bounding volumes in the MIWD Phase

1 test so choosing one over the other was not seen as a big contributory factor. In the tests in this chapter, the Hod bounding volume was chosen.

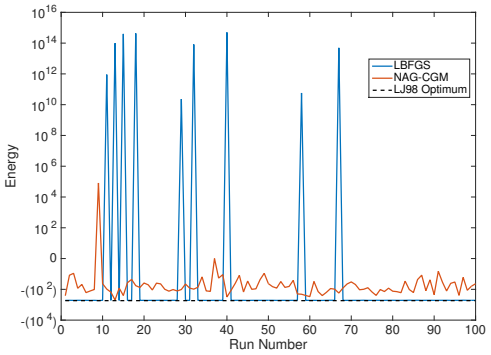
4.2 NAG CGM versus L-BFGS for Hod Bounding Volume

Figure 4.1 shows the comparison of results for NAG CGM and L-BFGS under the Hod bounding volume for test clusters $LJ_{77,98,102-104}$. Relaxed configurations for L-BFGS, generally resulted into lower energies compared to NAG CGM. L-BFGS, however, showed large spikes of energies in some of its runs while NAG CGM results were consistently clustered within 2-3 orders of magnitude higher than the global optima for all test clusters. The configurations associated with the energy spikes in the plots were investigated and showed to have particles in the core situated very close to each other causing the repulsive term of the potential energy funtion to dominate. The rest of the results of L-BFGS (excluding the spikes), however, were significantly lower in energies than any of the results found in its NAG CGM counterpart. L-BFGS generated lower relaxed energies 99% of the time for LJ_{102} and LJ_{103} while 94% for LJ_{77} and 91% for both LJ_{98} and LJ_{104}. The significant performance of L-BFGS among all test clusters is a good indication of the superiority of this relaxation method. Looking at the plots excluding the few extreme values generated by L-BFGS (See Figure 4.3 for a closer look) and comparing them again with NAG CGM, the former is a reliably consistent relaxation method. It is to be noted that although the plots may seem like L-BFGS results achieved the global optimum, they are all suboptimal. This is an indication that steps need to be further done to traverse the potential energy surface from the resulting configurations.

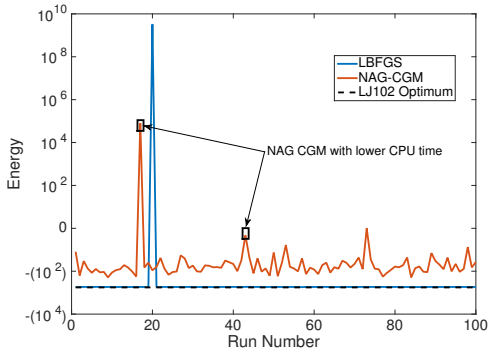
The time it took to relax the clusters were also recorded for both NAG CGM and L-BFGS (Figure 4.2). The program used the *clock()* function of the C programming language to record the CPU time in seconds. Looking at the CPU time for NAG CGM from LJ_{77} to LJ_{104} (increasing cluster size), an expected increase in amount of CPU time can be seen. This is not necessarily the case for the recorded CPU times under the L-BFGS relaxation method. CPU times under the LJ_{77} is generally lower however the recorded times for the rest of the test clusters do not appear to increase as cluster size increases. The recorded times from LJ_{98} to LJ_{104} is between 0 to < 2 seconds with mean < 1 second for all test clusters. NAG CGM showed more variability and longer CPU times from between 0 to 250 seconds with mean at most 16 seconds for LJ_{77} and at most 38 seconds for LJ_{104}. The difference in CPU time between L-BFGS and NAG CGM is very significant and will not translate well when thousands of iterations/runs



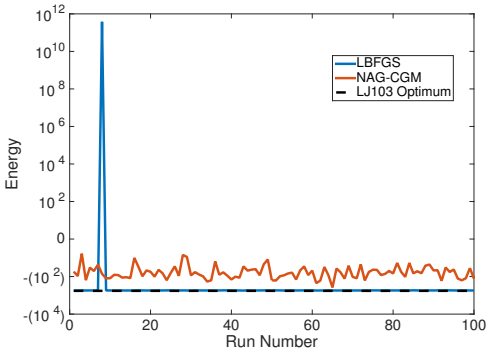
(a) LJ77



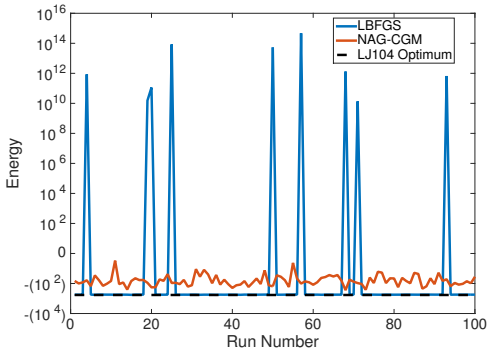
(b) LJ98



(c) LJ102



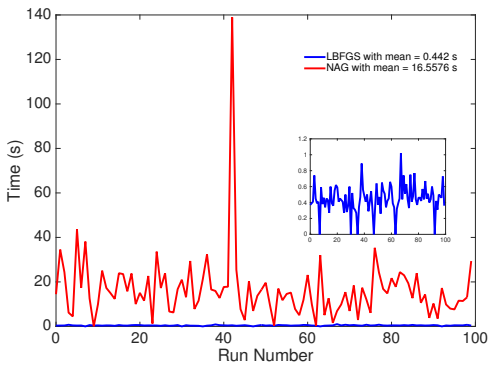
(d) LJ103



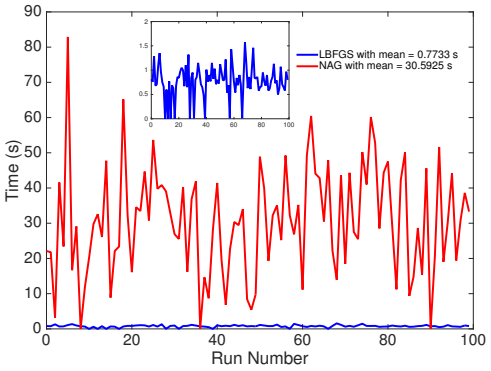
(e) LJ104

Figure 4.1: Comparison of relaxed energies using the Hod bounding volume for all test clusters. L-BFGS, although generated a few very high relaxed energies, generally outperformed NAG CGM across all test clusters.

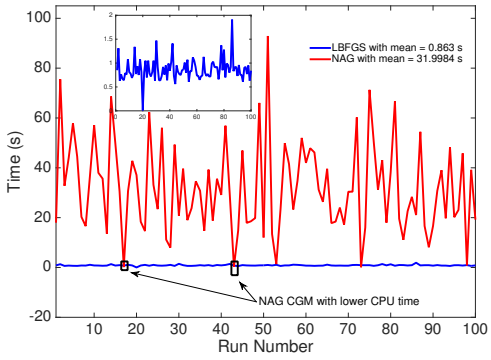
need to be done specially for larger clusters. A couple of runs under the LJ_{102} test cluster, however, showed faster relaxation time for NAG CGM (Figure 4.2c). Mapping these run numbers to its corresponding energies in Figure 4.1c show that L-BFGS energies are significantly lower.



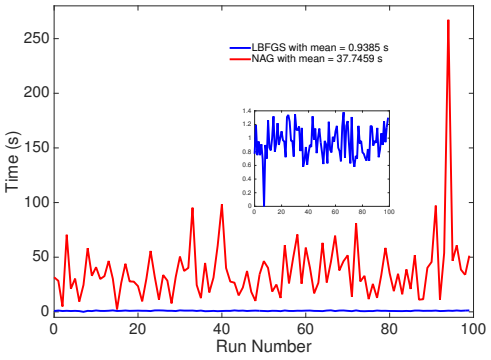
(a) LJ77



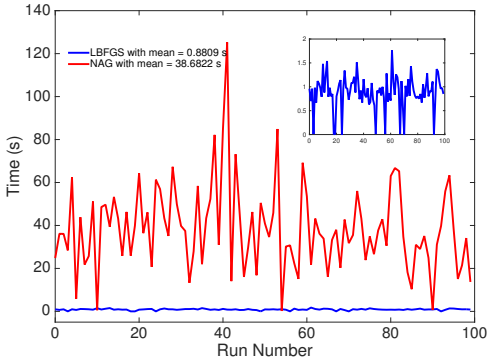
(b) LJ98



(c) LJ102



(d) LJ103

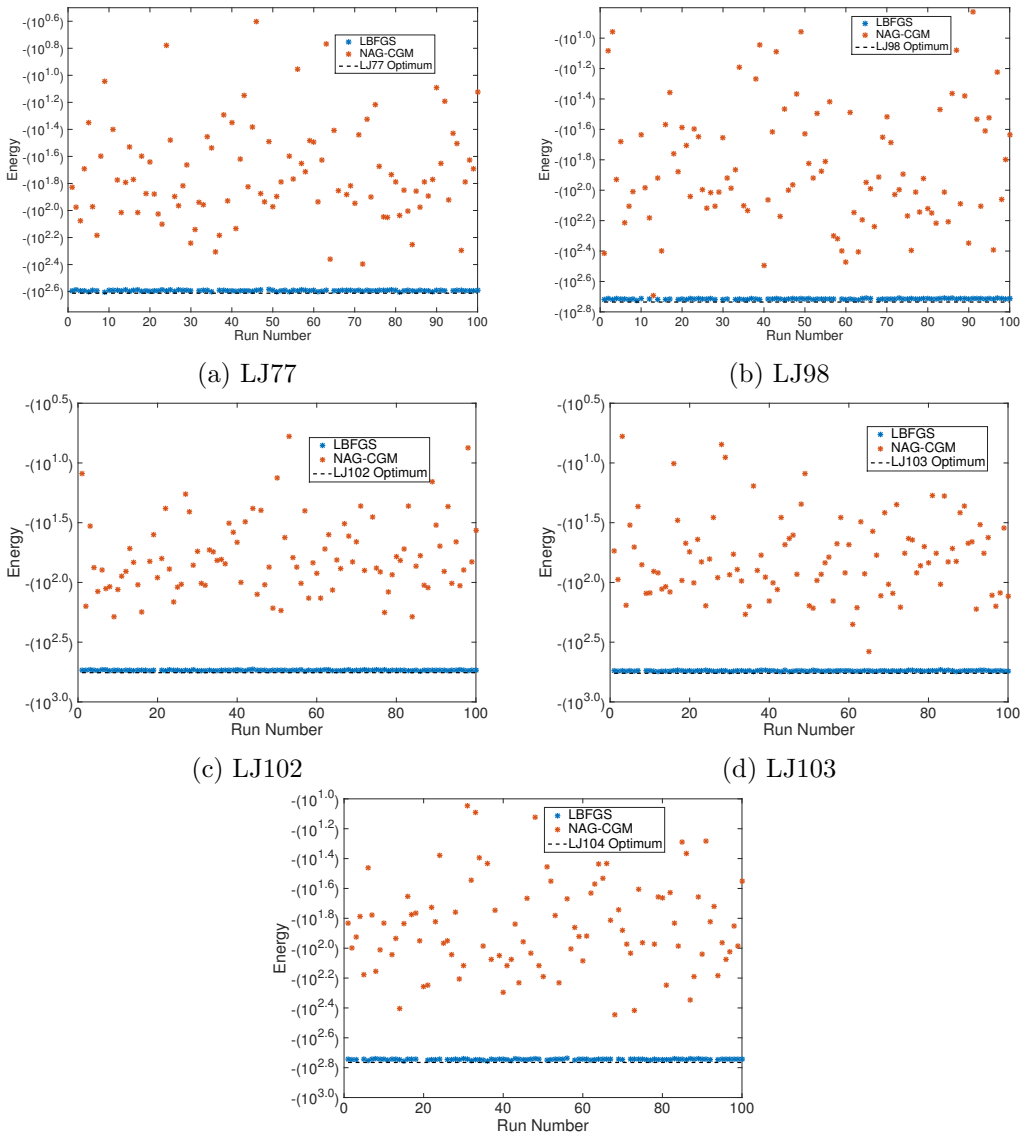


(e) LJ104

Figure 4.2: Recorded CPU time for relaxing configurations generated using the Hod bounding volume for all test clusters. L-BFGS CPU time in all test clusters were exceptionally faster than NAG CGM with a mean of less than 1 second across all test clusters while NAG CGM generated a mean of a little over 38 seconds in the largest test cluster of LJ_{104}.

4.3 NAG CGM versus L-BFGS for Wales Bounding Volume

Figure 4.4 shows the results from the runs comparing NAG CGM and L-BFGS under the Wales bounding volume for test clusters $LJ_{77,98,102-104}$. Similar to the observations of comparisons between NAG CGM and L-BFGS under the Hod



(a) LJ77

(b) LJ98

(c) LJ102

(d) LJ103

(e) LJ104

Figure 4.3: Relaxed energies comparing NAG CGM and L-BFGS excluding extreme values under Hod bounding volume for all test clusters. A good separation of energies can be seen between NAG CGM and L-BFGS relaxed energies.

bounding volume, the L-BFGS relaxed energies generally showed lower energies. This time, however, more extreme values (spikes) can be observed in all test clusters resulting in lower percentage of performance of L-BFGS than in the Hod case. The percentage performance of L-BFGS having lower energies than NAG CGM for test clusters LJ_{77}, LJ_{98}, LJ_{102}, LJ_{103}, and LJ_{104} are 87%, 76%, 70%, 79%, and 75%, respectively. Figure 4.6 show the plots of relaxed energies excluding the extreme values generated by L-BFGS. Unlike the results from Hod bounding volume, there is no clear separation of relaxed energies between NAG

CGM and L-BFGS under this bounding volume however it was still obvious that the latter generally obtained lower energies in all test clusters. The energies in this bounding volume is also observed to have more lower energies compared to the Hod bounding volume for all test clusters.

There was a single run in NAG CGM under the LJ_{103} test cluster when relaxation time was faster (Figure 4.5d). Mapping this run number with its corresponding energy in Figure 4.4d shows that L-BFGS, however, still generated lower relaxed energy.

4.4 Hod versus Wales bounding volume under L-BFGS

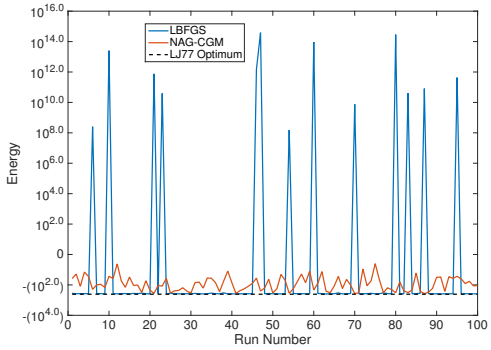
Looking closely at the relaxed energies between Hod and Wales under the L-BFGS relaxation method (Figure 4.7), Hod bounding volume generally located lower relaxed energies than Wales across test clusters. This has also been observed in a previous experiment when tested on MIWD Phase 1.

4.5 Summary and Conclusion

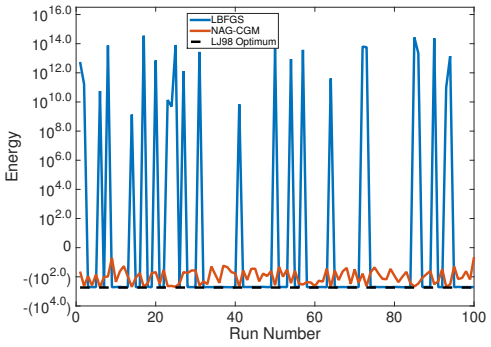
The ability of NAG CGM and L-BFGS to relax configurations have been tested on two bounding volumes, Hod and Wales, for 5 special test clusters namely, LJ_{77}, LJ_{98}, LJ_{102}, LJ_{103}, and LJ_{104}. Both relaxed potential energies and CPU time were compared for the two relaxation methods. Relaxed energies excluding extreme values under the L-BFGS method for Hod and Wales were also compared.

Results show that L-BFGS was an exceptional relaxation method both in terms of relaxed potential energies and CPU time. There is an indication that CPU time increases as cluster size increases for NAG CGM whereas the increase in cluster size did not affect L-BFGS CPU time at all across test clusters. Hod and Wales bounding volumes under L-BFGS showed that the latter generally relaxes to higher energies than the former. This agrees with the observations when tested on MIWD Phase 1 of the algorithm from 3.

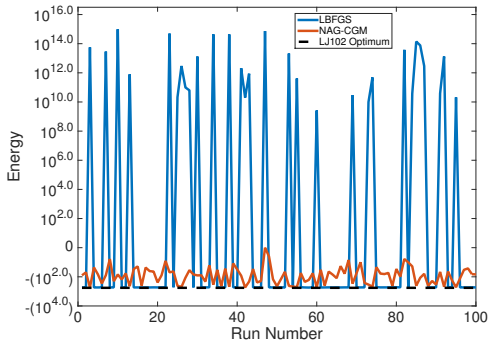
Using the results in this experiment and the previous chapter, the final runs of the MIWD algorithm will use Hod bounding volume with which to scatter initial particle sites while using L-BFGS as the relaxation method.



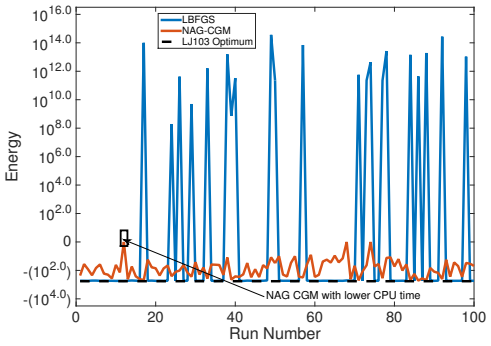
(a) LJ77



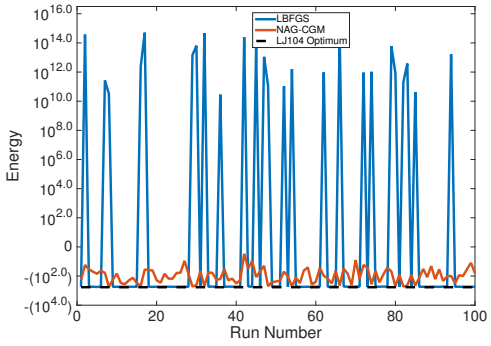
(b) LJ98



(c) LJ102

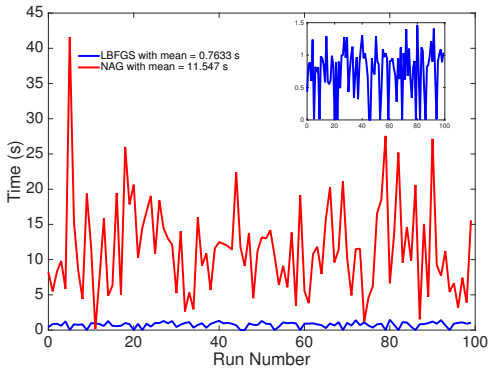


(d) LJ103

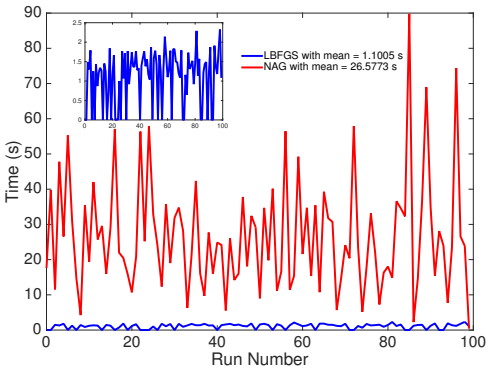


(e) LJ104

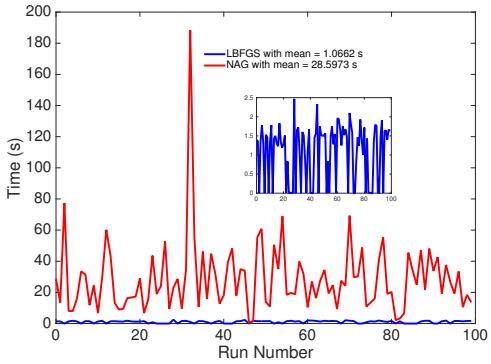
Figure 4.4: Comparison of relaxed energies using the Wales bounding volume for all test clusters. L-BFGS, although generated a few very high relaxed energies, generally outperformed NAG CGM across all test clusters. A similar observation seen when using Hod bounding volume. The spikes in the L-BFGS in this bounding were evidently higher than found in Hod bounding volume.



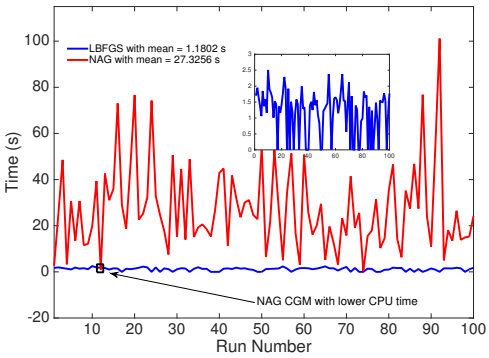
(a) LJ77



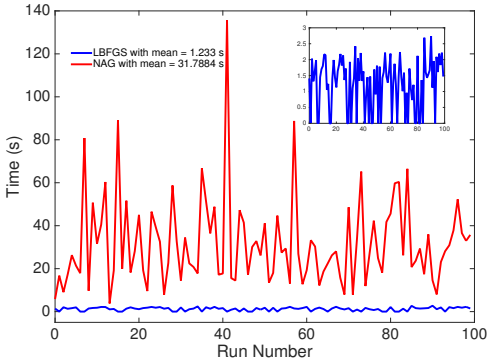
(b) LJ98



(c) LJ102

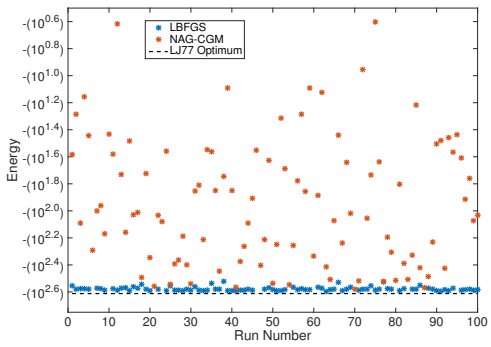


(d) LJ103

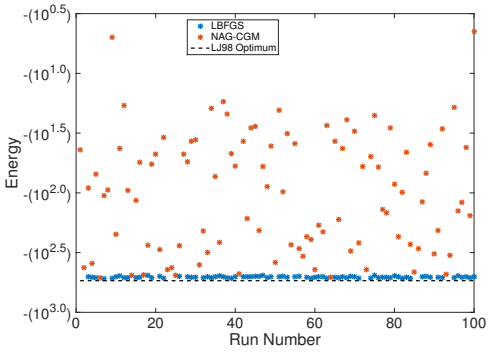


(e) LJ104

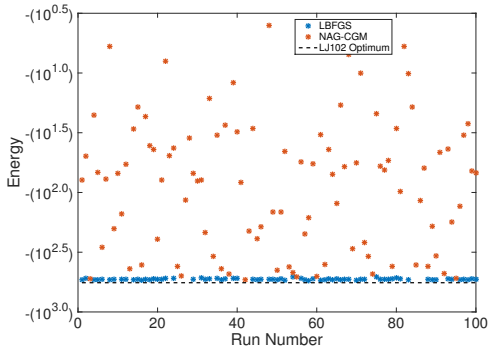
Figure 4.5: Recorded CPU time for relaxing configurations generated using the Wales bounding volume for all test clusters. L-BFGS CPU time in all test clusters were exceptionally faster than NAG CGM with a mean of less than 1.3 seconds at its worst across all test clusters while NAG CGM generated a mean of a little over 0.5 min in the largest test cluster of LJ_{104}.



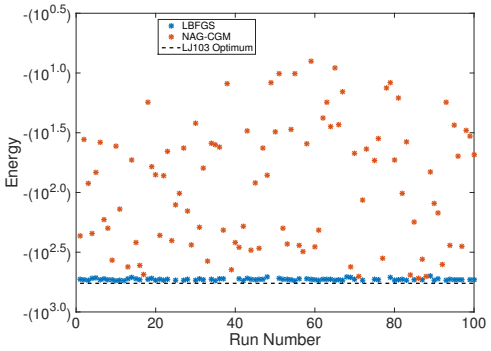
(a) LJ77



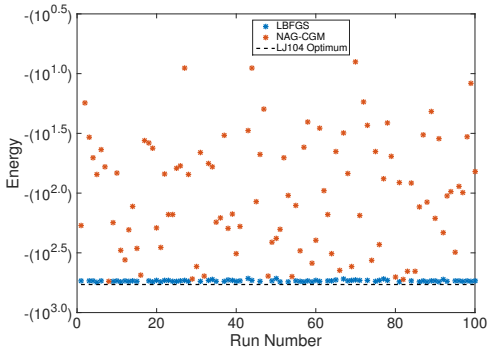
(b) LJ98



(c) LJ102

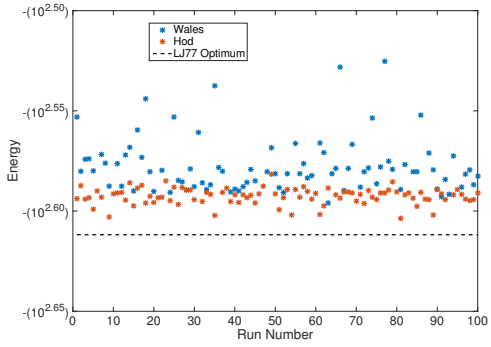


(d) LJ103

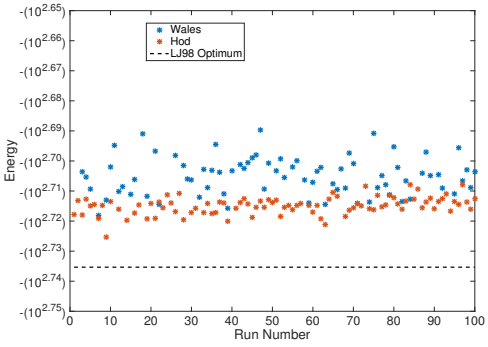


(e) LJ104

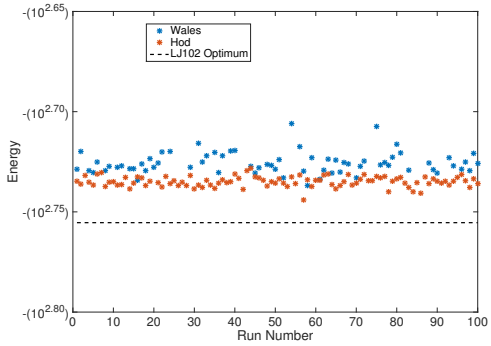
Figure 4.6: Relaxed energies comparing NAG CGM and L-BFGS excluding extreme values under Wales bounding volume for all test clusters. There are more lower energies for the NAG-CGM relaxation method using this bounding volume compared to the Hod bounding volume.



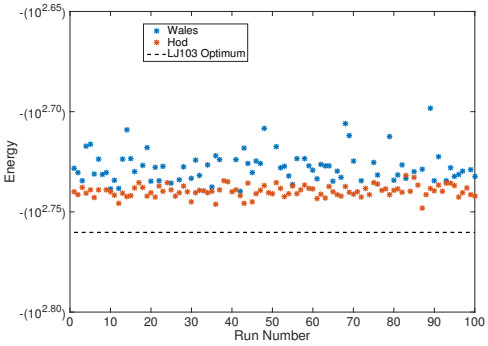
(a) LJ77



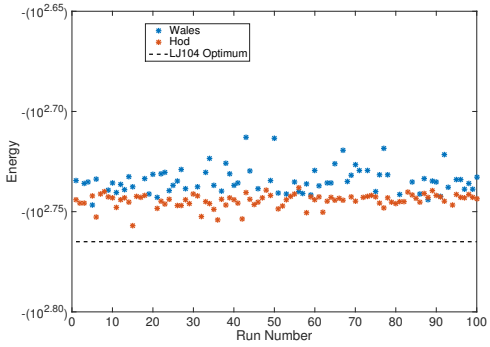
(b) LJ98



(c) LJ102



(d) LJ103



(e) LJ104

Figure 4.7: Comparison of relaxed energies using L-BFGS between Hod and Wales bounding volumes excluding extreme values. Hod bounding volume generally showed lower relaxed energies in all test clusters.

Chapter 5

Perturbation Operators

5.1 Introduction

Test runs of the Modified Intelligent Water Drops (MIWD) Phase 1 on test clusters LJ_{38} and LJ_{98} showed that relaxed configurations of the final iterations could still not locate the proper basin that contains the global optima. This chapter shows test run results of selected perturbation operators, detailed in chapter 2, to further fine-tune results generated by Phase 1 of MIWD as tested on test clusters LJ_{13} and LJ_{38}. The addition of these perturbation operators comprises the Phase 2 of the MIWD+PerOp algorithm. The plots in the succeeding sections show the global minima line, the energies of the relaxed configuration at the end of Phase 1, and the selected iterations from the start until the end of Phase 2. The aim of this test is mainly to observe the behaviour of the different perturbation operators and determine which combinations of bounding volume and operator will be used for the final runs of MIWD+PerOp.

5.2 Operator Tests on LJ_{13}

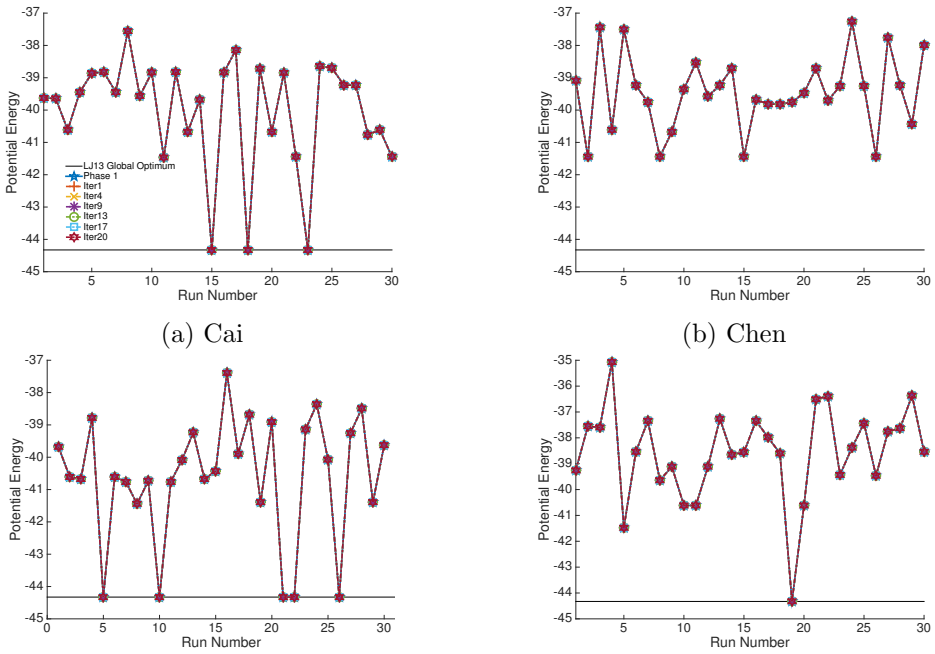
For LJ_{13}, MIWD Phase 1 was iterated for 1000 iterations and Phase 2 for 20 times using 30 indepedent runs. Figures 5.1 to 5.9 show the results of all ten perturbation operators. In each of the figures, the global minimum for LJ_{13} at -44.326801ε is shown by a horizontal black solid line. The sections below detail the performance of each operator.

5.2.1 Geometric and Arithmetic Mean

Figures 5.1 and 5.2 show the results for Geometric and Arithmetic Mean perturbation operators, respectively. A small percentage of Phase 1 results were able to hit the global minimum under the Cai, Hod and Wales bounding volumes but

application of both operators, across all bounding volumes, performed poorly. Perturbing and relaxing the non-global optimum results in Phase 1 did not improve the energies thus showing no decrease in energy in any of the iterations in Phase 2.

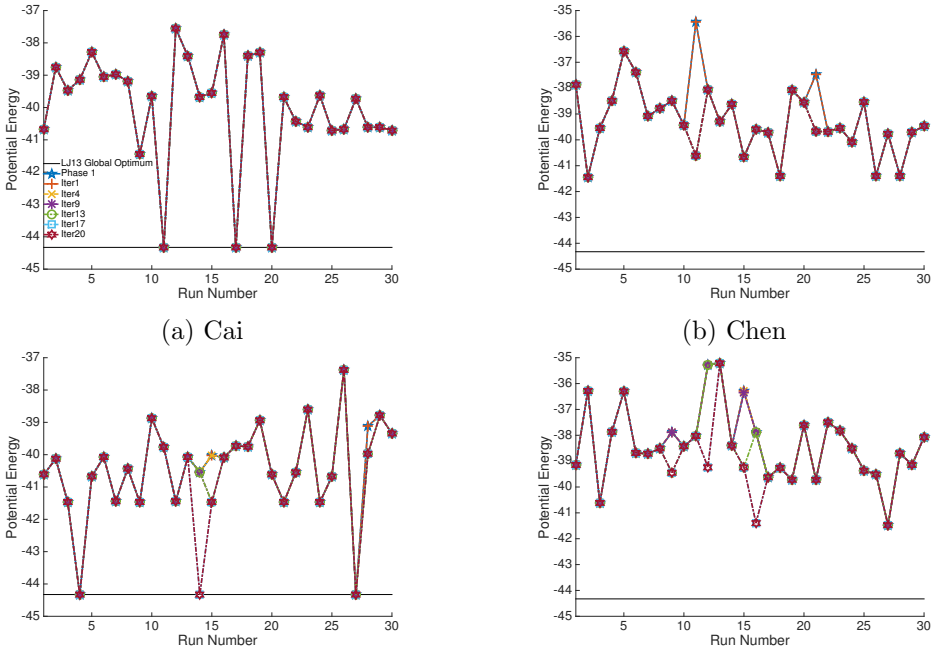
The poor performance of both of these operators can be attributed to the fact that the operations involved in generating a new candidate cluster could be very disruptive to the overall geometry of the cluster. This drastic change to the geometry could be causing it to move away further from the basin of the GO thus not allowing it to find lower energies than the current one.



(a) Cai (b) Chen

(c) Hod (d) Wales

Figure 5.1: Comparison of search trajectory of energies using MIWD with Geometric Mean Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}. The plot key *Phase 1* is the plot referring to the relaxed energy at the end of Phase 1 while the plot keys *Iter1* to *Iter20* are plots of energies at the end of the Phase 2 iteration associated with the number.



(a) Cai

(b) Chen

(c) Hod

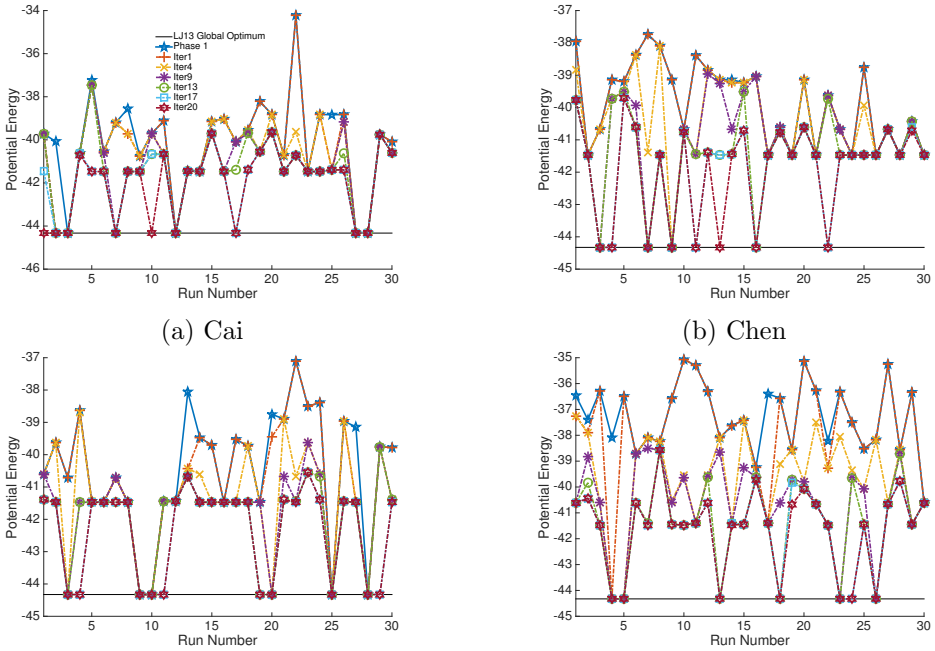
(d) Wales

Figure 5.2: Comparison of search trajectory of energies using MIWD with Arithmetic Mean Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

5.2.2 Laplace Crossover and Power Mutation

Figures 5.3 and 5.4 show the results for Laplace Crossover and Power Mutation, respectively. Laplace Crossover generates a candidate cluster by a displacement of positions of all the particles of one parent cluster. The displacement will be based on a factor of the difference between the corresponding particle positions of one parent cluster and another. The factor, which takes a value within $(-\infty, 1.0]$ or $[1.0, +\infty)$, uses a value based on a Laplace Distribution. The generation of new clusters using Power Mutation is very similar to Laplace Crossover except that the displacement, which could be significantly smaller than Laplace Crossover, will be based on a factor of the difference between the particle position of a parent cluster and a fixed lower or upper limit. The factor, limited within $(0.0, 1.0]$, is a value based on a power function.

Looking at how this operator performed in terms of hitting the global optimum, Table 5.1 shows the percentage of successes for each operator on the different bounding volumes. The table, and the succeeding tables hereafter, do not take into account the clusters with optimal energies at the end of Phase 1. The higher percentage success rates for Power Mutation further shows that a more "fine-tuned"operator could generate lower energies.



(a) Cai

(b) Chen

(c) Hod

(d) Wales

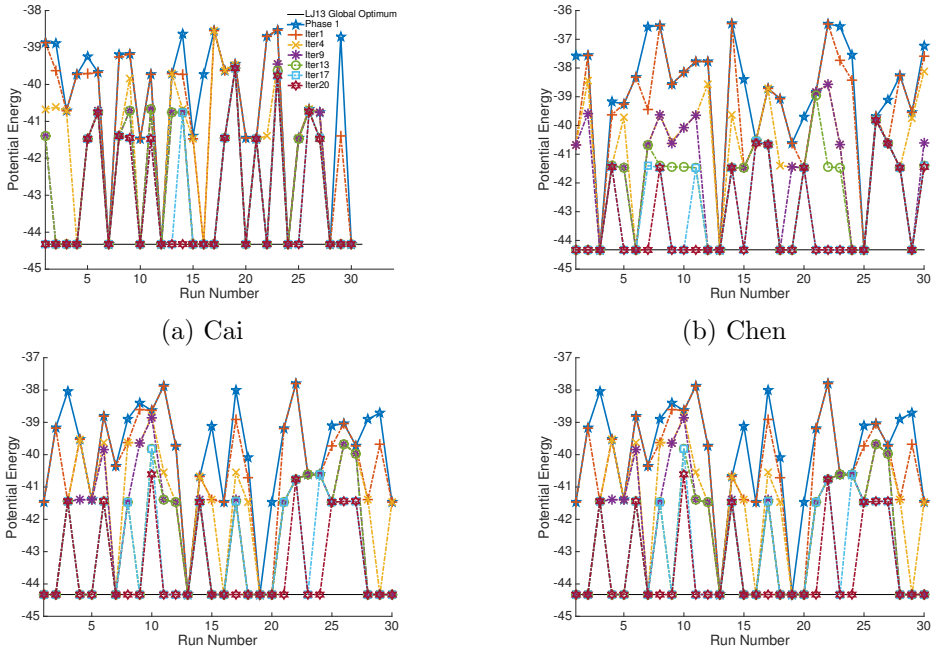
Figure 5.3: Comparison of search trajectory of energies using MIWD with Laplace Crossover Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

Table 5.1: Percentage success for the Laplace Crossover and Power Mutation perturbation operators for LJ_{13}.

	Bounding Volume	Improved clusters
Laplace Crossover	Cai	19%(5/27)
	Chen	27%(8/30)
	Hod	27%(7/26)
	Wales	23%(7/30)
	Bounding Volume	Improved clusters
Power Mutation	Cai	54%(13/24)
	Chen	63%(17/27)
	Hod	71%(20/28)
	Wales	79%(23/29)

5.2.3 Two-Point and N-point Crossover

Figures 5.5 and 5.6 show the results for Two-point and N-point perturbation operators, respectively. These operators generate new clusters by essentially copying certain particle positions from each of its parents. The difference between the two is the number of queries made by the operator. In N-point crossover, particles in the new cluster will equally-likely get either of the parent's particles. In Two-point crossover, a group of selected particles in one parent is exchange with a



(a) Cai

(b) Chen

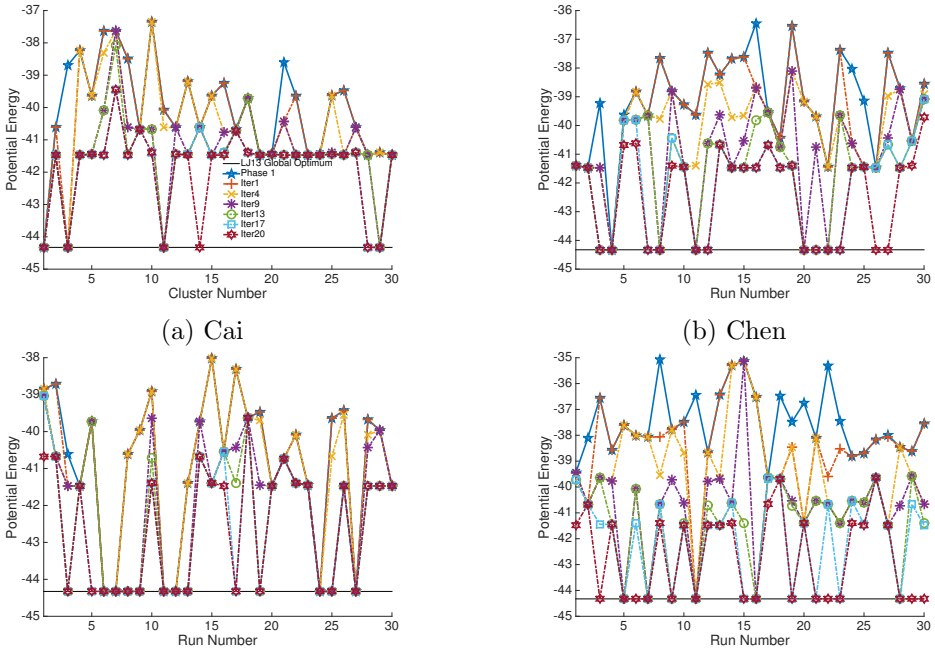
(c) Hod

(d) Wales

Figure 5.4: Comparison of search trajectory of energies using MIWD with Power Mutation Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

group of selected particles from another parent. These operators are less disruptive if both parents have very similar geometries but could drastically generate new clusters when they are very dissimilar. These operators are implemented in code using an (linear) array of values representing positions of particles. In N-point crossover, the operator goes through each set of values in the array and queries whether to get the particle for that position from parent 1 or parent 2. In the worst case scenerio (degree of change in geometry), the child cluster will get on its alternate array position a particle from different parents. For Two-point crossover, a group of contiguous particle positions from the array representation of parent 1 (e.g. particles in position 1 to k of parent 1) is concatenated with another contiguous particle positions from parent 2 (e.g. particles from position $k+1$ to n - number of particles in cluster - of parent 2). The worst case scenario for Two-point crossover is when the cut point is in the middle of the array.

Table 5.2 shows the percentage of success of the two operators for each bounding volume. Between the two very similar operators, the Two-point crossover is significantly better than N-point crossover in all bounding volumes except for Cai. This result is expected as N-point crossover has the tendency to be more disruptive to the geometry than Two-point crossover.



(a) Cai (b) Chen

(c) Hod (d) Wales

Figure 5.5: Comparison of search trajectory of energies using MIWD with Two-point Crossover Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

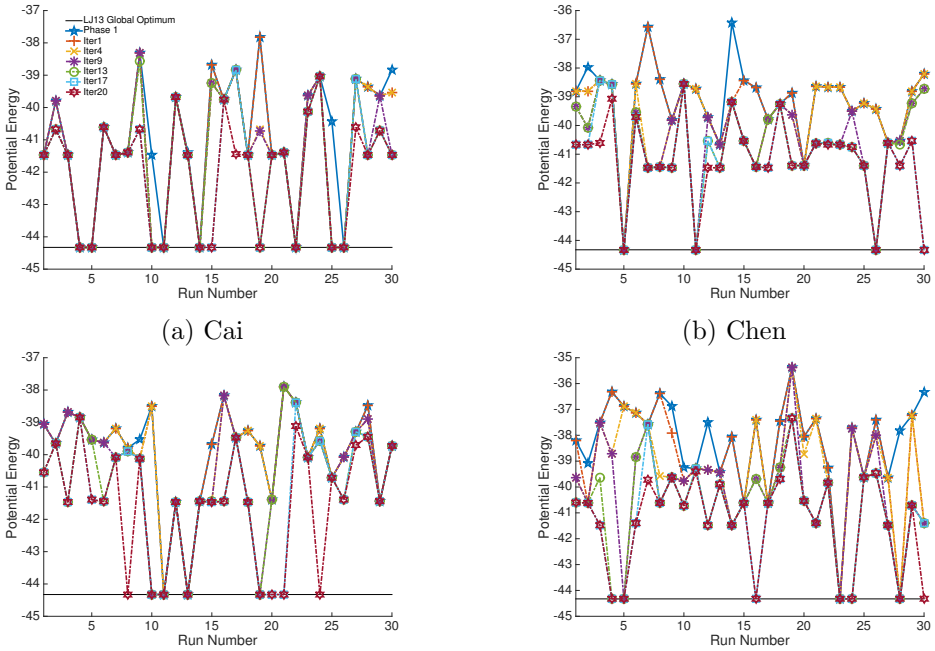
Table 5.2: Percentage success for the Two-point and N-Point Crossover perturbation operators for LJ_{13}.

	Bounding Volume	Improved clusters
Two—Point Crossover	Cai	17%(5/29)
	Chen	38%(11/29)
	Hod	33%(8/24)
	Wales	50%(15/30)
	Bounding Volume	Improved clusters
N—Point Crossover	Cai	17%(4/24)
	Chen	10%(3/29)
	Hod	21%(6/28)
	Wales	21%(6/29)

5.2.4 Twinning

Figure 5.7 shows the results for Twinning. Twinning spatially rotates, by a random angle, a group of particles belonging to a randomly chosen side of a randomly chosen plane.

Table 5.3 shows the percentage of success of this operator for each bounding volume. Hit success rates for this operator across all bounding volume is a significant improvement even for Cai bounding volume which performed relatively



(a) Cai (b) Chen

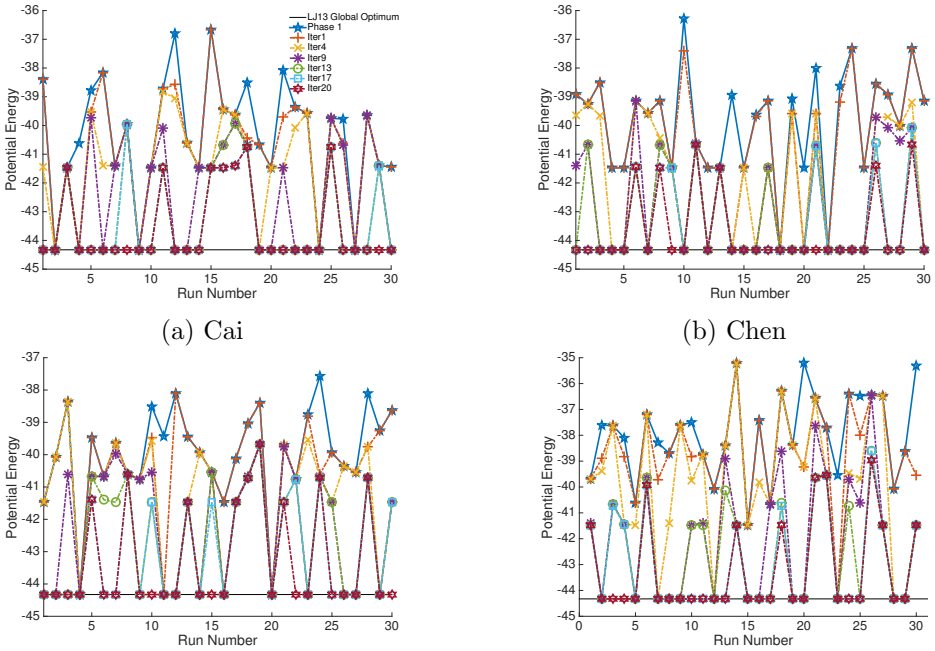
(c) Hod (d) Wales

Figure 5.6: Comparison of search trajectory of energies using MIWD with N-point Crossover Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

poor in the other operators presented so far.

Table 5.3: Percentage success for the Twinning perturbation operator for LJ_{13}.

Bounding Volume	Improved clusters
Cai	73%(19/26)
Chen	79%(22/28)
Hod	67%(18/27)
Wales	70%(21/30)



(a) Cai

(b) Chen

(c) Hod

(d) Wales

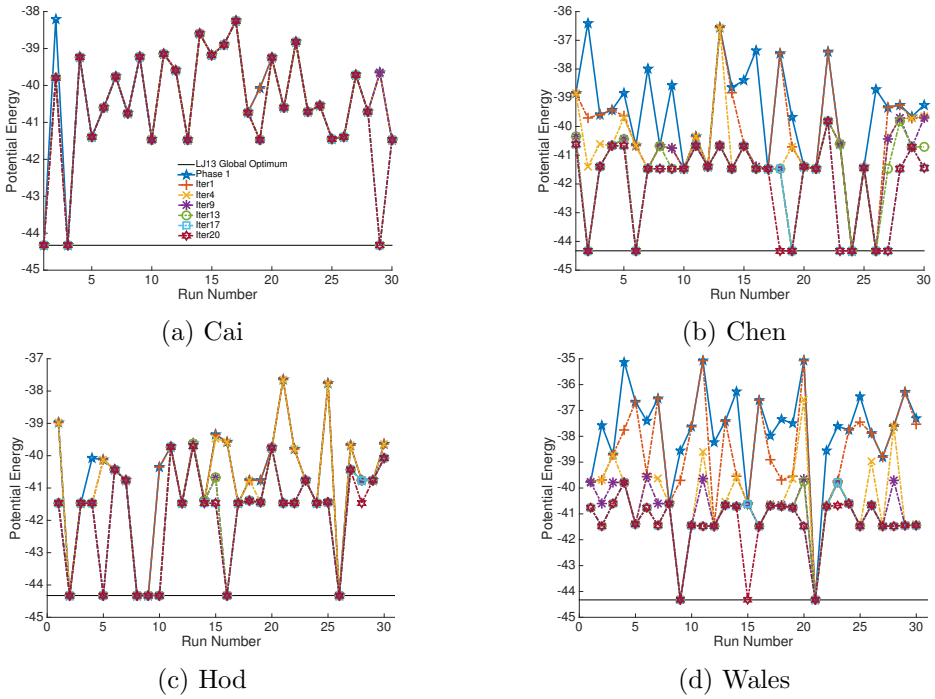
Figure 5.7: Comparison of search trajectory of energies using MIWD with Twinning Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

5.2.5 Etch—and—Grow and Grow—and—Etch

Figures 5.8 and 5.9 show the results for Etch—and—Grow and Grow—and—Etch, respectively. The Etch—and—Grow operator removes high-energy particles before gradually adding particles within the volume of the current cluster (the original implementation adds particles to the surface of the etched cluster). Grow—and—Etch, on the other hand, is the opposite of Etch—and—Grow, but adds particles within the volume of the cluster before gradually etching high-energy particles one by one. This implementation slightly differs from the original operator where the growing step is only concentrated on the surface of the cluster. These operators are considered less disruptive compared to the two previous operators so far as each addition only makes small adjustments to the geometry of the cluster and there is a good chance that previous neighbouring particles will still be retained after relaxation. Both these operators' "fine-tuning" characteristic allowed it to find lower energies for the suboptimal clusters in Phase 1. This can be clearly seen by how separated the blue-star and maroon-hexagram plotlines are in all the figures.

Table 5.4 show the percentage of success rates for the two operators for each bounding volume. Grow—and—Etch was able to hit a 100% success rates for Chen, Hod and Wales bounding volumes. Etch—and—Grow, on the other hand,

faired poorly in comparison with Grow—and—Etch with none of the bounding volumes getting more than 25% percentage success rate. This performance confirms the remark in a previous study [Wolf & Landman, 1998] where these operators were originally applied. This comparison shows that implementing the "etching" step at the end rather than at the start is more advantageous in finding lower energies.



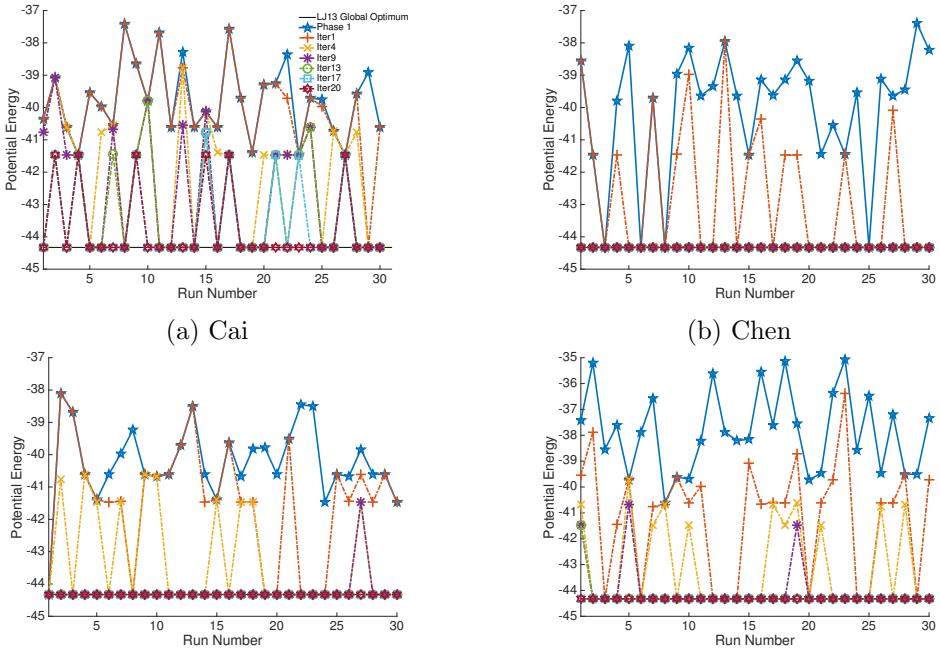
(a) Cai (b) Chen

(c) Hod (d) Wales

Figure 5.8: Comparison of search trajectory of energies using MIWD with Etch and Grow Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

Table 5.4: Percentage success for the Etch—and—Grow and Grow—and—Etch perturbation operators for LJ_{13}.

	Bounding Volume	Improved clusters
Etch—and—Grow	Cai	4%(1/28)
	Chen	24%(7/29)
	Hod	12%(3/26)
	Wales	7%(2/29)
	Bounding Volume	Improved clusters
Grow and Etch	Cai	80%(24/30)
	Chen	100%(26/26)
	Hod	100%(29/29)
	Wales	100%(30/30)



(a) Cai

(b) Chen

(c) Hod

(d) Wales

Figure 5.9: Comparison of search trajectory of energies using MIWD with Grow and Etch Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

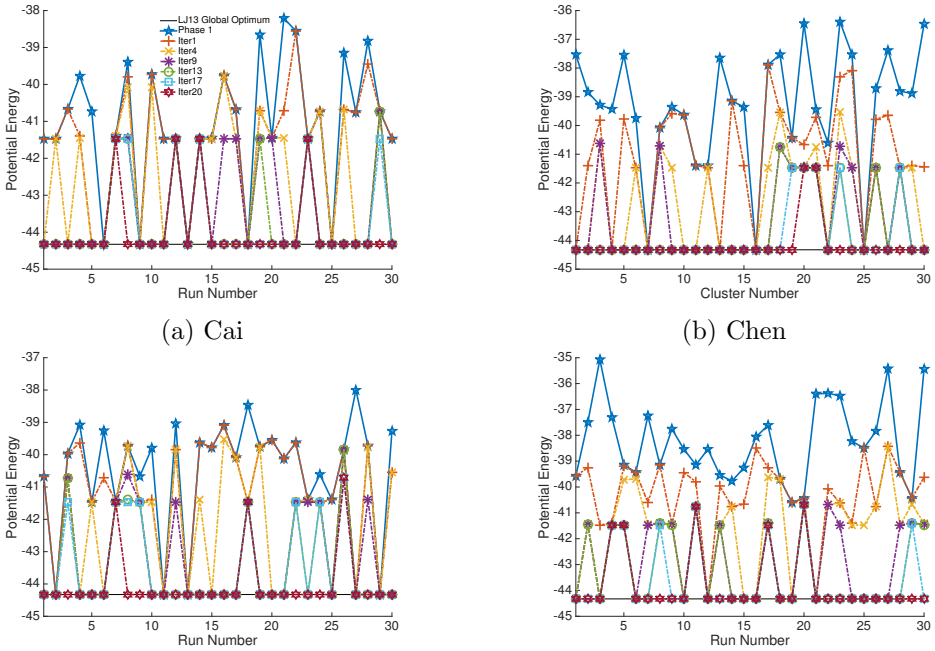
5.2.6 Inversion

Figure 5.10 shows the results for Inversion. A cluster is represented in code as a (linear) array of particle positions and "inverting" simply flips a randomly selected portion of the array to generate a new cluster. This operator can introduce more change to the geometry the longer the length of the selected portion. However, since the displacement is within the values of the current particle positions and only affects the inverted portion, this operator is not considered extremely disruptive to the geometry.

Table 5.5 show the percentage of success of this operator for each bounding volume. Improvements from Phase 1 to Phase 2 across all bounding volumes are impressively significant and can be considered the next best perturbation operator to Grow—and—Etch.

Table 5.5: Percentage success for the Inversion perturbation operator for LJ_{13}.

Bounding Volume	Improved clusters
Cai	84%(21/25)
Chen	93%(25/27)
Hod	85%(23/27)
Wales	83%(25/30)



(a) Cai

(b) Chen

(c) Hod

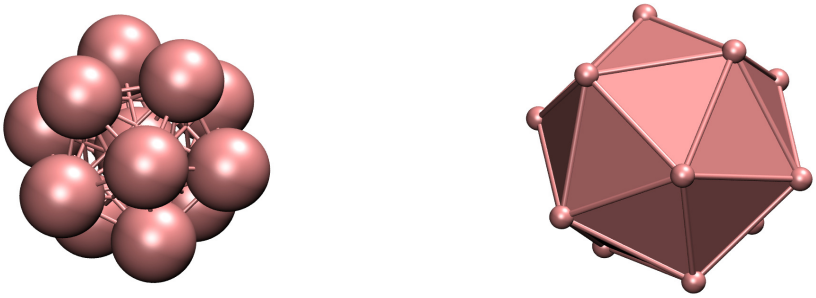
(d) Wales

Figure 5.10: Comparison of search trajectory of energies using MIWD with Inversion Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{13}.

5.2.7 Overall Performance of Operators on LJ_{13}

The results above show that Geometric and Arithmetic Mean perturbation operators could not provide improvement to energies of clusters generated from Phase 1 of MIWD. The remaining operators, Etch—and—Grow, Laplace Crossover, Two—point and N—point Crossover, Power Mutation, Twinning, Inversion and Grow—and—Etch, were able to show varying degrees of improvements to MIWD Phase 1 suboptimal clusters. Among these, the operators Power Mutation, Twinning, Inversion and Grow—and—Etch, showed higher success hit rates in finding better geometries or lower energies. The success rate for the best operators range from 54% to 100%. It was also observed that more geometry—disrupting operators are likely to perform poorly than more fine—tuning types of operators. Among all perturbation operators, only Grow—and—Etch operator was able to hit the 100% success rate in most of the bounding volumes tested. Across all perturbation operators, the Cai bounding volume still had difficulty in competing with the rest of the bounding volumes. Both Chen and Hod bounding volumes performed relatively at par with each other.

Figure 5.11 show the optimal structure for LJ_{13} as generated by Grow—and—Etch using the Hod bounding volume.



(a) Ball and stick representation of the LJ_{13} GO.

(b) Polyhedral (icosahedron) representation of the LJ_{13} GO. Each vertex represents the center of a particle.

Figure 5.11: GO structure for LJ_{13} generated generated using MIWD+PerOp with Grow—and—Etch as the perturbation operator and using the Hod bounding volume.

5.3 Operator Tests on LJ_{38}

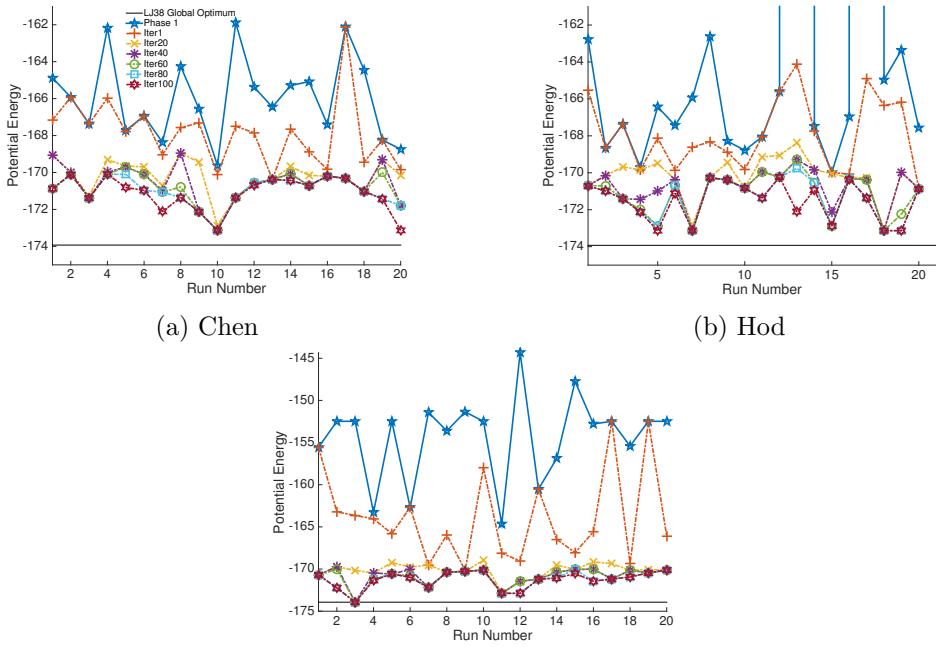
LJ_{13} can be considered as a relatively trivial test cluster to optimize thus the perturbation operators are also tested on a larger and considered more difficult LJ cluster, the LJ_{38}. LJ_{38} is a special LJ cluster because of its truncated octahedral structure which provides a stiff test for any global optimization algorithm. For this test, only 20 runs were done with MIWD Phase 1 increased to 2000 iterations and Phase 2 increased to 100 iterations. Furthermore, only 4 perturbation operators (Power Mutation, Twinning, Inversion and Grow—and—Etch) and 3 bounding volumes (Chen, Hod and Wales) were used in this test. These were selected based on the performance of these operators and bounding volumes in the LJ_{13} case. In each of the figures, the global minimum for LJ_{38} at -173.928427ε is shown by a horizontal black solid line.

5.3.1 Overall Performance of Operators on LJ_{38}

Figures 5.12, 5.13, 5.14 and 5.15 show the results for Power Mutation, Twinning, Inversion and Grow—and—Etch under the LJ_{38} test cluster, respectively. None of the results from MIWD Phase 1 were able to generate the GO. A few results at the end of MIWD Phase 1 obtained very high energies (e.g. Hod in Figure 5.12, Hod and Wales in Figure 5.13, Hod in Figure 5.14, and Chen and Wales in Figure 5.15) but these were not included in the plots to provide a better look at the energies of the iterations in MIWD Phase 2. Iteration plot lines for MIWD Phase 2 clearly shows capability of the selected perturbation operators to find lower energies. The sucess rate, however, at the end of the 100th iteration for

operator-bounding volume combinations were not satisfactory. The hit success is at most 2 out of 20 and the worst without any hit at all. Being one of the most difficult LJ clusters, it not surprising that these operators could not hit the GO in 100 iterations.

Looking at the average final energies for each bounding volume in each of the operators, Power Mutation and Twinning rank third with energies in the range -171.94ε to -171.10ε, Inversion is next with a range between -172.85ε to -172.60ε and Grow—and—Etch as the best with a range between -173.90ε to -173.09ε.



(a) Chen

(b) Hod

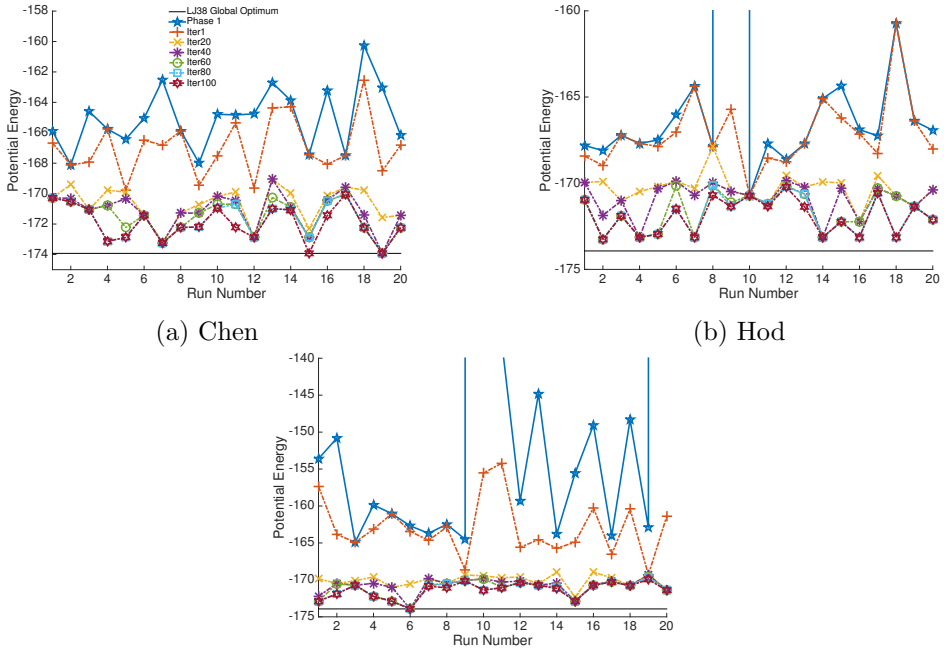
(c) Wales

Figure 5.12: Comparison of search trajectory of energies using MIWD with Power Mutate Perturbation Operator for Chen, Hod and Wales bounding volumes tested using LJ_{38}.

Figure 5.16 show the optimal structure for LJ_{38} as generated by Grow—and—Etch using the Hod bounding volume.

5.4 Summary and Conclusion

Ten different perturbation operators and bounding volumes were tested on LJ_{13}. Potential energies were compared and percentage success rates were recorded. It was found out that the operators Power Mutation, Twinning, Inversion and Grow—and—Etch performed relatively well among the ten operators on the bounding volumes Chen, Hod and Wales. It was observed that more geometry—



(a) Chen

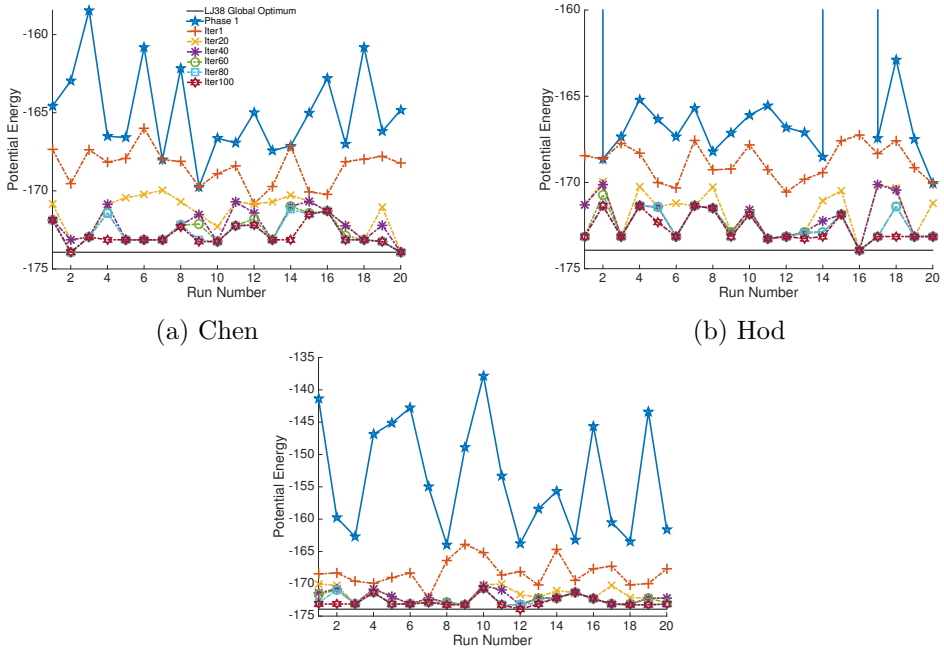
(b) Hod

(c) Wales

Figure 5.13: Comparison of search trajectory of energies using MIWD with Twinning Perturbation Operator for Cai, Chen, Hod and Wales bounding volumes tested using LJ_{38}.

disruptive operators perform worse than more 'fine—tuning' operators which are characteristics of the mentioned best performing operators.

The best performing operators and bounding volumes were further tested on LJ_{38}. Although hit rates were poor, this was to be expected as only 100 iterations were done on Phase 2 and further supports the requirement that longer iterations may need to be done as cluster size increases. Among the four operators, Grow—and—Etch is the most promising and will be used as the perturbation operator in the final runs of this study. Together with the Hod bounding volume and L—BFGS, which was shown to be a better bounding volume and relaxation method combination in chapter 4, MIWD+PerOp (MIWD Phase 1 and Phase 2) can now be applied to the LJ system up to size 104.

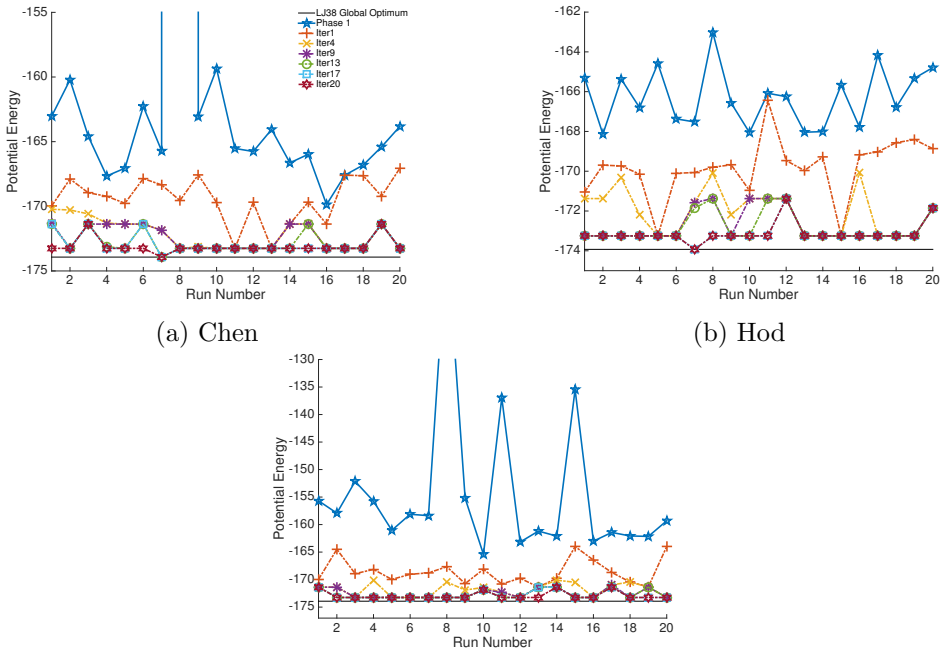


(a) Chen

(b) Hod

(c) Wales

Figure 5.14: Comparison of search trajectory of energies using MIWD with Inversion Perturbation Operator for Chen, Hod and Wales bounding volumes tested using LJ_{38}.

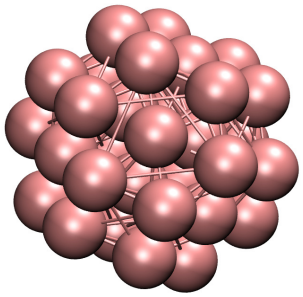


(a) Chen

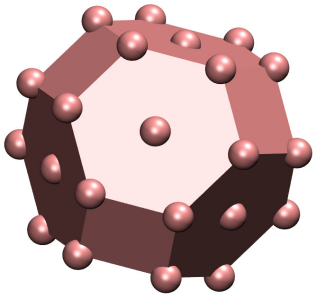
(b) Hod

(c) Wales

Figure 5.15: Comparison of search trajectory of energies using MIWD with Grow and Etch Perturbation Operator for Chen, Hod and Wales bounding volumes tested using LJ_{38}.



(a) Ball and stick representation of the LJ_{38} GO.



(b) Polyhedron (truncated octahedron) representation of the LJ_{38} GO. Each vertex represents the center of a particle.

Figure 5.16: GO structure for LJ_{38} generated using MIWD+PerOp with Grow—and—Etch as the perturbation operator and using the Hod bounding volume.

Chapter 6

MIWD+PerOp for Lennard-Jones Cluster Optimization

6.1 Introduction

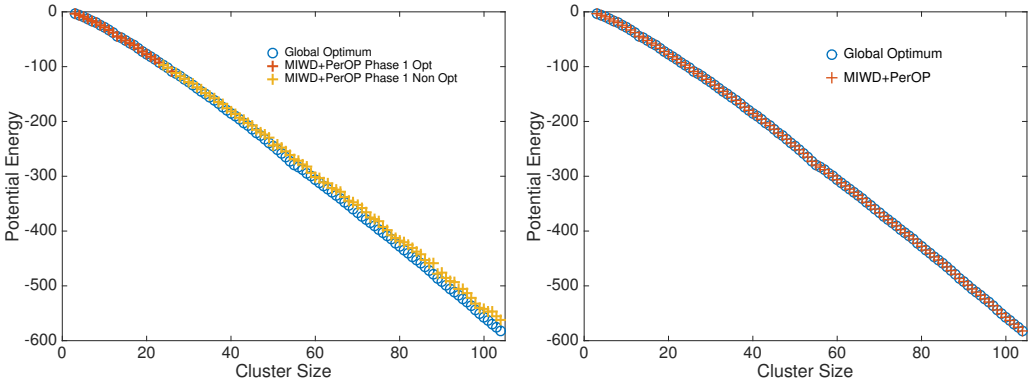
Results in chapter 3, chapter 4 and chapter 5 provided us information on: (1) the capability of MIWD+PerOp to find lower energies from purely random configurations; (2) L-BFGS as the better relaxation method option; and (3) the best combination of bounding volume and perturbation operator to use. Using these information, this chapter will summarize the results of the application of MIWD+PerOp to LJ clusters up to size 104. It is worth mentioning this early that all putative global optima for LJ clusters sizes $3 \leq N \leq 104$ atoms reported in the Cambridge Cluster Database [Wales *et al.*, n.d.] were found in this study.

6.2 Results

Initially 40 independent runs were used for cluster sizes 3 to 9, however, this was decreased to 20 independent runs for cluster sizes 10 to 104 which proved to be an adequate number and time efficient. A run is interpreted as one pass of MIWD+PerOp steps 1 to 14.

We first look at how MIWD+PerOp performed after Phase 1. Figure 6.1a compares the energies obtained after Phase 1 of MIWD+PerOp and the putative global optima. MIWD+PerOp Phase 1 alone was able to obtain the global optima for LJ cluster sizes 3 to 23 and 26, thus, no further optimization (i.e. no application of Phase 2 or perturbation operators) was done. This performance could probably be improved with increased number of particle sites during the

initialization step of the algorithm as this would allow the algorithm to sample more combinations of particles thereby potentially traversing a wider potential energy surface. However, it has to be noted that increasing the number of particle sites in Phase 1 requires more computational effort and memory usage as more probabilities would have to be computed. This study did not explore that option. For the rest of the LJ cluster sizes with suboptimal potential energies at the end of Phase 1, Phase 2 of MIWD+PerOp was run. All putative global optima were located at the end of Phase 2 without exception (Figure 6.1b).



(a) MIWD (Phase 1) final energies compared to the LJ putative global optima. Red crosses indicate hitting global optimum at the end of Phase 1 while Yellow crosses indicate sub-optimal energies.

(b) MIWD+PerOp potential energies compared to the putative global optima

Figure 6.1: MIWD+PerOp and CCD LJ putative global optima comparison for Lennard-Jones Clusters.

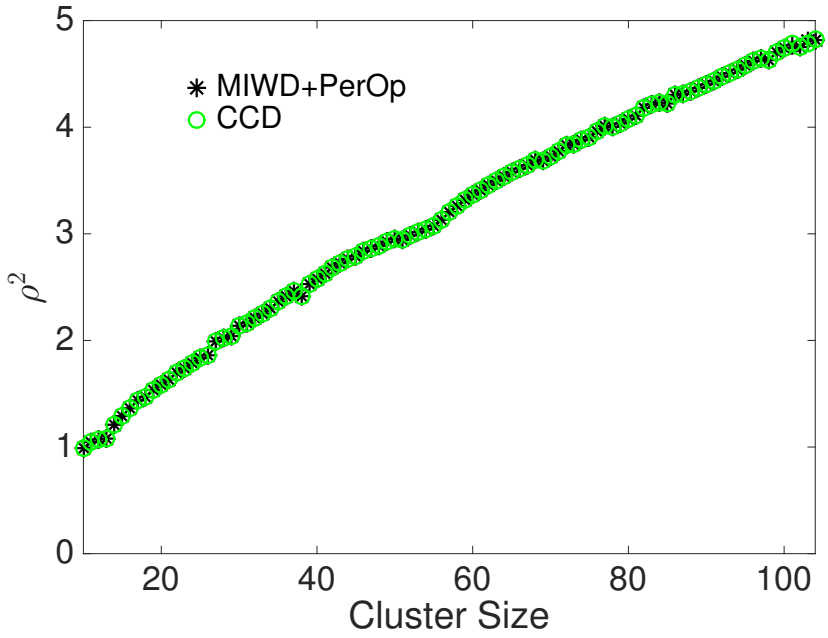
Measures to characterize the geometry of the clusters were calculated for the results of this study similar to what was done in Marques, et al [Marques & Pereira, 2010]. The compactness of a cluster was determined by calculating the squared hyperradius, ρ^2 (Equation 6.1). The compactness of the clusters (Figure 6.2) in this study agree exactly with the results in CCD.

$$\rho^2 = (I_1 + I_2 + I_3)/(2M_n) \tag{6.1}$$

where I_1, I_2, and I_3 are the moments of inertia and M_N is the total mass. In this study, a particle is taken as 1 mass unit.

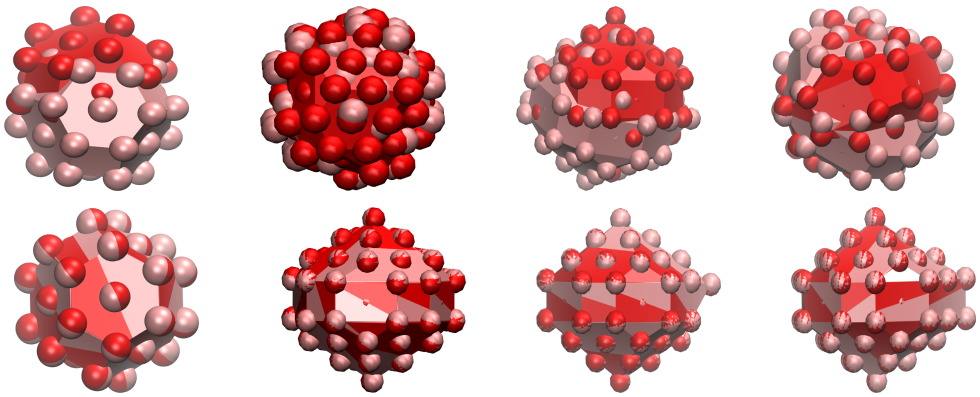
CCD and MIWD+PerOp cluster configurations were overlaid using VMD [Humphrey *et al.*, 1996] to check how well particle positions match. Rotations and translations done for the clusters in our study show that it is able to match the configurations of the results in CCD. Figures 6.3 and 6.4 show some of the clusters rotated and translated to match the geometries of clusters in CCD.

Figure 6.5 shows the success rates obtained on each cluster size. A run is



(a)

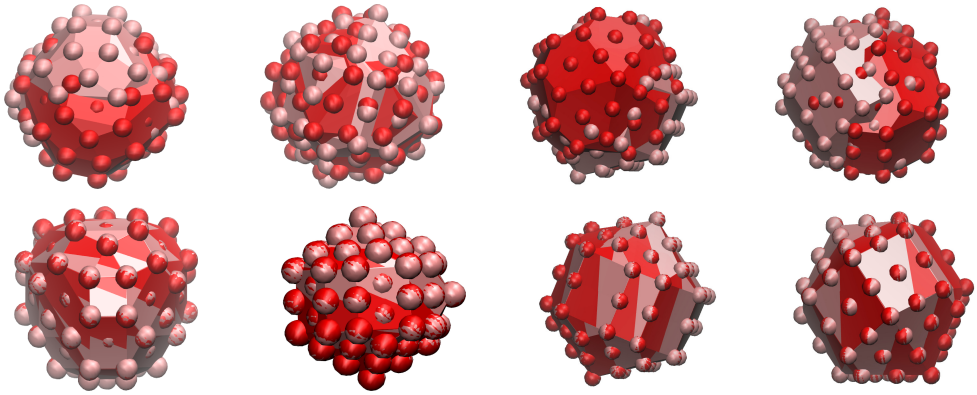
Figure 6.2: Comparison of compactness for the different LJ cluster sizes between MIWD+PerOp and CCD results. Compactness values calculated using the squarred hyperradius, ρ^2, for the MIWD+PerOp results matches with CCD results for all sizes tested without exception.



(a) LJ38 (b) LJ75 (c) LJ76 (d) LJ77

Figure 6.3: Clusters in red are CCD configurations while clusters in salmon are MIWD+PerOp results. Top row : Overlaid clusters showing unmatched particle positions. Bottom row : Rotated and translated clusters showing matching particle positions.

considered successful when, at the end of the run, the best IWD is able to find the putative global optimum for that cluster size. Most clusters tested achieved 100% success rates out of the total number of runs. The test clusters with less than 50% success rates came from the most difficult clusters characterized as Mark



(a) LJ98 (b) LJ102 (c) LJ103 (d) LJ104

Figure 6.4: Clusters in red are CCD configurations while clusters in salmon are MIWD+PerOp results. Top row : Overlaid clusters showing unmatched particle positions. Bottom row : Rotated and translated clusters showing matching particle positions.

decahedra except for LJ_{83} (45% success rate) and LJ_{101} (25% success rate) . The Marks decahedra classified clusters are LJ_{76}, LJ_{77}, LJ_{102}, LJ_{103}, and LJ_{104}.

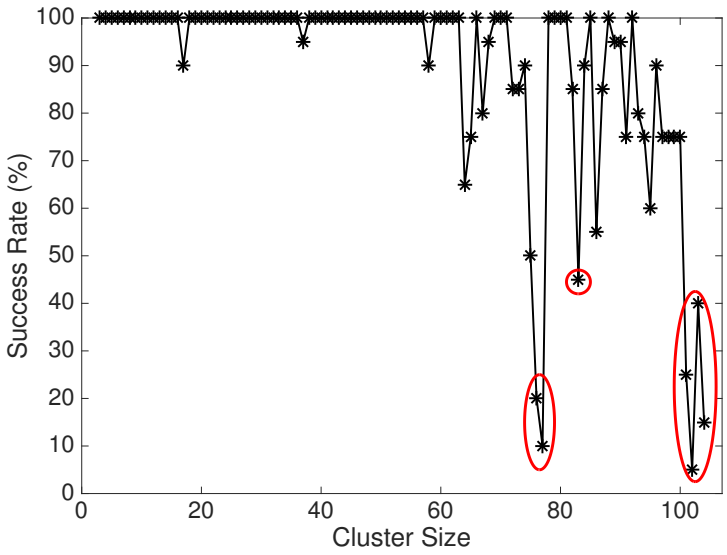


Figure 6.5: MIWD+PerOp success rates for the LJ clusters up to size 104. Tests mostly generated 100% success rates except for LJ_{76}, LJ_{77}, LJ_{83}, LJ_{101}, LJ_{102}, LJ_{103}, and LJ_{104}.

In a study using Basin-Hopping [Wales & Doye, 1997], the lowest energy configuration for LJ_{78} was found from a seeded run using the lowest energy configuration of LJ_{79}. In our study, LJ_{78} achieved a 100% success rate without any seeding. Furthermore, global optima for LJ_{76}, LJ_{77}, LJ_{103} and LJ_{104} were also achieved using seeded runs in Basin-Hopping while all these difficult clusters were found using unbiased initial configurations in this study.

Comparing MIWD+PerOp to algorithms which were tested on small LJ clusters such as FAEA [Cai & Shao, 2002] and PSO [Hodgson, 2002], this study was able to successfully discover the putative global minima for cluster sizes LJ_3 to LJ_{13} in 100% of all the runs where FAEA only achieved 100% on LJ_2 to LJ_6 and 95% on LJ_7 to LJ_{13}. PSO suffered high failure rates even for small LJ clusters and failed to find the global optima altogether for LJ_{13}. Furthermore, the capability of MIWD+GrowEtch in achieving high success rates for difficult clusters is also proven against the original Basin-Hoping, its effective extension called Basin-Hopping with Occasional Jumping (BHOJ) [Iwamatsu & Okabe, 2004], Monotonic Sequence Basin-Hopping (MSBH) [Leary, 2000] and Parallel Fast Annealing Evolutionary Algorithm [Cai *et al.*, 2002a]. Table 6.1shows the significant differences in success rates among these methods. It is worth mentioning that MIWD+PerOp was able to get as high as 50% success rate for LJ_{75} despite its reputation as a difficult case which is evident in the low success rates obtained in other methods. With the exception of LJ_{102}, MIWD+PerOp beat all other methods.

Table 6.1: Percentage success of MIWD+GrowEtch vs BH, BHOJ, MSBH and PFAEA. With the exception of LJ_{102}, MIWD+GrowEtch outperformed several basin hopping variants and an annealing-based algorithm.

Cluster Size	MIWD+ PerOp	BH	BHOJ	MSBH	PFAEA
38	100%	87%	96%	12.4%	39%
75	50%	1%	5%	4%	1%
76	20%	5%	10%	0.8%	4%
77	10%	6%	5%	0.2%	2%
98	75%	10%	10%	0.6%	4%
102	5%	3%	16%	3.10%	9%
103	40%	3%	13%	no data	10%
104	15%	3%	12%	no data	7%

¡

6.3 Summary and Conclusion

A Modified IWD with Grow Etch operator was successfully applied to LJ clusters of sizes 3 to 104. Algorithm runs showed that Phase 1 of the algorithm was sufficient to locate the putative global optima of cluster sizes from 3 to 23 atoms and 26 atoms. For size 25 atoms and LJ clusters sizes up 27 up to 104 atoms, Phase 1 results provided good configurations which were useful as unbiased starting points for further optimization using the Grow Etch perturbation operator.

Configurations generated in this study agree exactly with the geometries found in CCD using comparisons of energies obtained, compactness of geometries and overlaying of configurations.

Multiple runs of the MIWD algorithm showed higher success rates than published results. A number of parameter values used in this study did not undergo parameter optimization, nevertheless, it did not affect its ability to find lower energies as iterations progress. The algorithm is clearly a promising alternative tool for energy minimization problems. The results of MIWD+PerOp for the LJ sytem builds a good foundation for its applicability to more complicated systems such as its heterogenous counterpart, the Binary LJ clusters, and a system known to have more diverse structural geometries called Morse Clusters. The succeeding chapters show the performance of MIWD on these systems. The algorithm was also used in predicting the lowest energies of Janus clusters in which particle interaction does not only rely on spatial properties but also on angular interactions.

Chapter 7

MIWD+PerOp and MIWD+CombiOp for Binary Lennard-Jones Cluster Optimization

7.1 Introduction

Binary clusters offer a more challenging task to the theoretician than its one-component case due to the presence of homotops and the additional variable of cluster composition that adds to the complexity of its structural behaviour [Doye & Meyer, 2005]. A binary cluster such as the Binary Lennard-Jones (BLJ)system hence has a combinatorial structure which provides a harder problem than the standard LJ case and has a much wider solution space [Cassioli *et al.*, 2009]. This system is thus a fitting test instance for MIWD+PerOp.

Several recent studies of the BLJ system locating the optimal proportion between atoms of type A and B has been successfully dealt with by their respective algorithms [Doye & Meyer, 2005, 2006;Marques & Pereira, 2010;Kolossvary & Bowers, 2010;Sicher *et al.*, 2011;Tao *et al.*, 2011]. In this study, however, an exhaustive analysis of the space of the compositional parameter is out of the question. The focus of the current work is on testing MIWD+PerOp for BLJ from size 5 up to size 50 with each cluster size tested for 6 instances of σ_{BB} (296 test instances all in all) and using the published optimal compositions of atom types as a fixed parameter. Although this provides a lesser challenge to a new optimization algorithm, BLJ still proves to be a more complex system than LJ due to the presence of isomers or clusters with the same geometric structure but differ in atom labeling. For consistency, we follow the same assumptions

used in Doye and co-workers [Doye & Meyer, 2005] and Cassioli and co-workers [Cassioli *et al.*, 2009] for the coupling coefficients: $\sigma_{AA} = 1.0$, $\sigma_{AB} = \frac{\sigma_{AA}+\sigma_{BB}}{2}$, $\varepsilon_{AB} = \varepsilon_{AA} = \varepsilon_{BB} = 1.0$. σ_{BB}/σ_{AA} varies in the range 1.0 to 1.30 for all clusters up to 50 atoms. This allows observation of how stable structures of the BLJ clusters change as atomic size ratio changes. The following sections show MIWD+PerOp using 4 different perturbation operators in Phase 2 of the algorithm. MIWD+CombiOp, which combines perturbation operators in a run, is also introduced in this chapter and implemented as part of the algorithm as a consequence of the shortcomings of MIWD+PerOp.

7.2 Results

MIWD+PerOp and MIWD+CombiOp for the BLJ system implemented different Phase 2 operators compared to its one-component counterpart. This is mostly influenced by the fact that the compositional aspect of the cluster needs to be preserved and simply displacing particle positions, as is the case in the one-component LJ system, would not allow better exploration of the Binary LJ potential energy surface. Four operators which are appropriate for a binary system were used, namely: Knead, Cut and Splice variant and two Energy-Based Swapping operators. Twenty-five independent runs were done for each cluster size, perturbation operator and σ_{BB} combination.

7.2.1 Energy-based Swapping Operators

The operators, called *Smooth* and *Flip* procedures, introduced by Tao and co-workers [Tao *et al.*, 2011] in their algorithm called *3OP* inspired two operators to be applied to BLJ as Phase 2 of the MIWD+PerOp algorithm. *Smooth* moves a high-energy type B particle to a most vacant site or low-energy site while *Flip* changes a particle type from A (B) to B (A). In this study, a combination of concepts from these two operators were used to develop *EBS_HALB* and *EBS_HBLA*. In these new operators, a high-energy type B particle is swapped in position with a low-energy type A particle (*EBS_HBLA*) and vice versa (*EBS_HALB*). Globally optimal BLJ clusters tends to clump together type A (smaller) particles into its core while type B (larger) particles envelope the core. These new operators are somewhat disruptive to the geometry as this breaks "shelling" of type A and B particles. Nevertheless, this is implemented with the hope of possibly escaping a locally suboptimal potential energy associated with that geometry.

Figures 7.1 and 7.2 show success rates for both operators up to size 20. Figures show that although relatively high success rates were observed for sizes 7 to 14 across σBB values, failure to hit the putative global optima (GO) were

observed for sizes 5, 6 and 15-20. The poor performance on low cluster sizes was a deciding factor to stop further testing on larger cluster sizes.

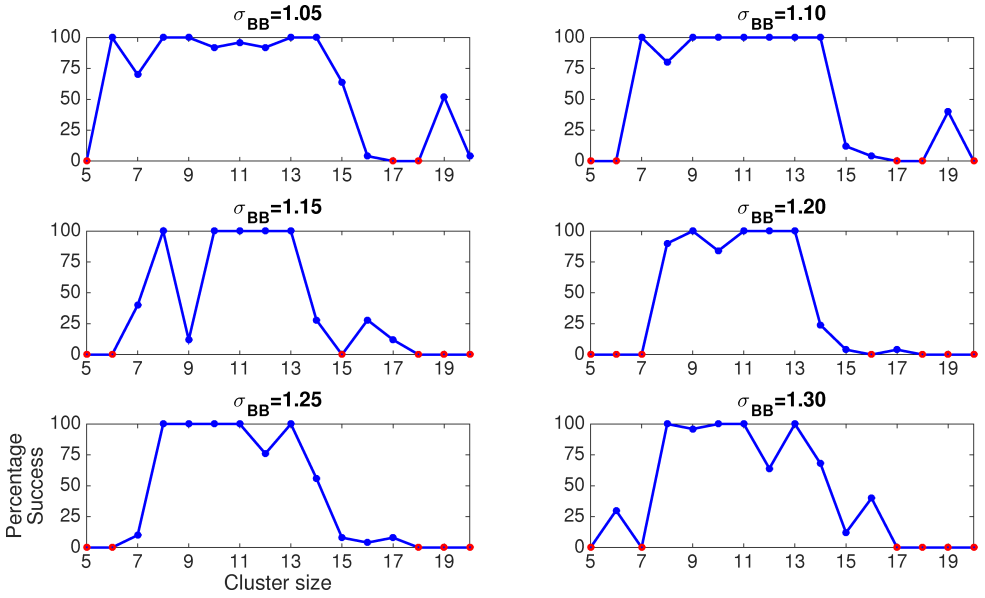


Figure 7.1: MIWD+PerOp success rates for the BLJ clusters up to size 20 using Energy-Based Swap HALB in Phase 2.

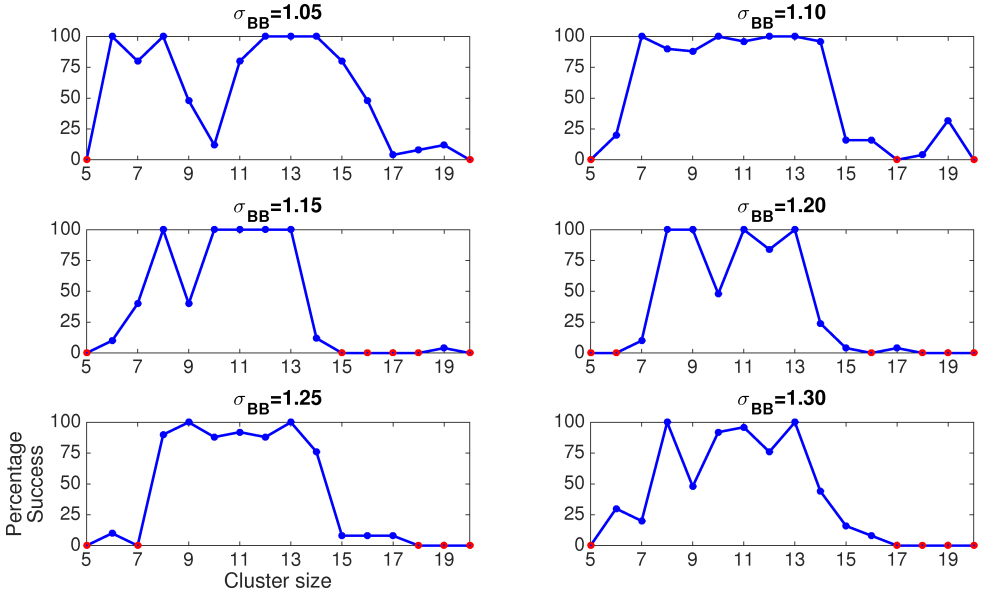


Figure 7.2: MIWD+PerOp success rates for the BLJ clusters up to size 20 using Energy-Based Swap HBLA in Phase 2.

7.2.2 Knead and CutSpliceVar

*Knead*ing as used in optimization of BLJ clusters is similar to bread kneading where the dough is massaged by taking a portion from the edge or outer part of the dough and folded into the centre. In this way, pockets of air are removed. In the same way, *kneading* in optimization selects particles which contribute the highest to the total energy and randomly places them into the centre of the cluster. The particles with the highest energies are usually found in the surface of the cluster due to the low number of nearest neighbors. The *Knead*ing procedure implemented in this study is similar to Tao and co-workers [Tao *et al.*, 2011].

Kneading shows better success rates (Figure 7.3) than the Energy-based operators. Although these success rates are low, Kneading was able to hit the putative GO for up to cluster sizes 50 with a few failures for cluster instances where type A and type B atoms are almost the same size ($\sigma_{BB} = 1.05$ and 1.10). Figure 7.4 shows the best energies generated by MIWD+PerOp with Knead as perturbation operator while Figure 7.5 compares these energies to the GO found in CCD. Figures with insets show the cluster instances with which MIWD+PerOp obtained suboptimal energies.

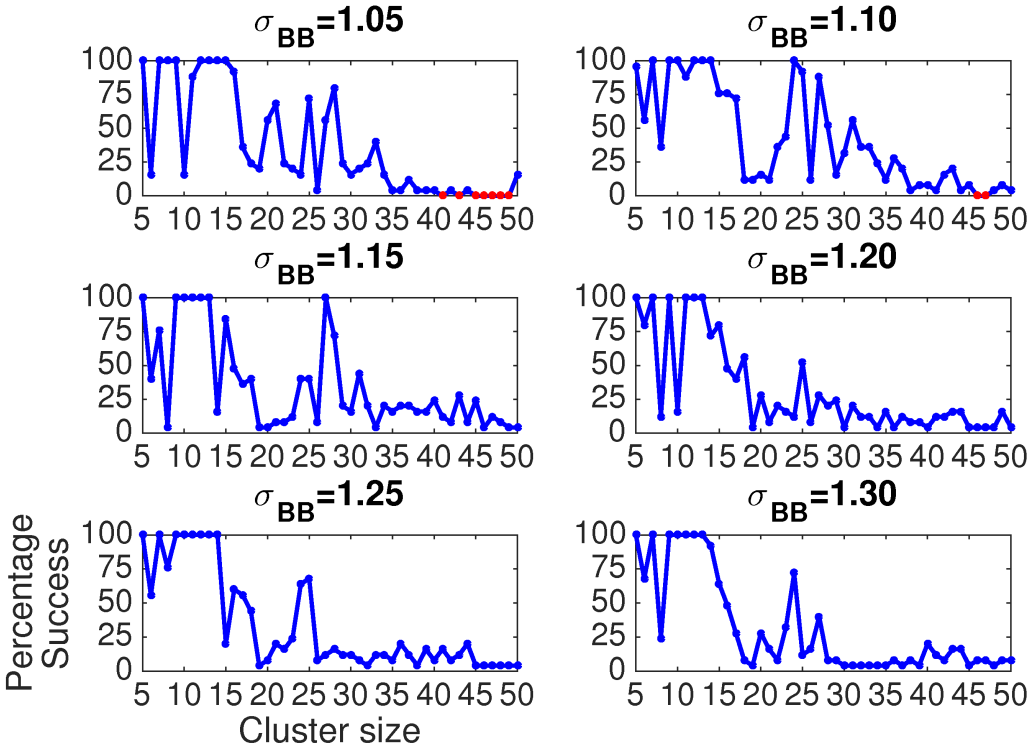


Figure 7.3: MIWD+PerOp success rates for the BLJ clusters up to size 50 using Kneading in Phase 2. Success rates as σ_{BB} increases declines for larger cluster sizes.

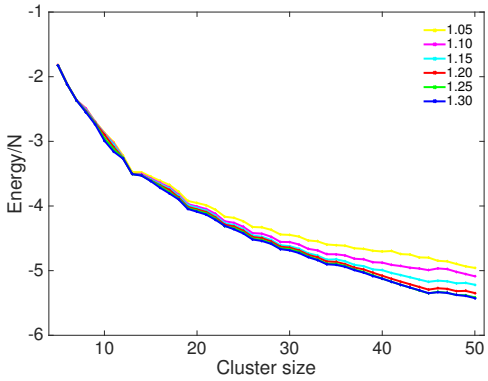


Figure 7.4: Energies generated by MIWD+PerOp for the different σ_{BB} using Knead as perturbation operator.

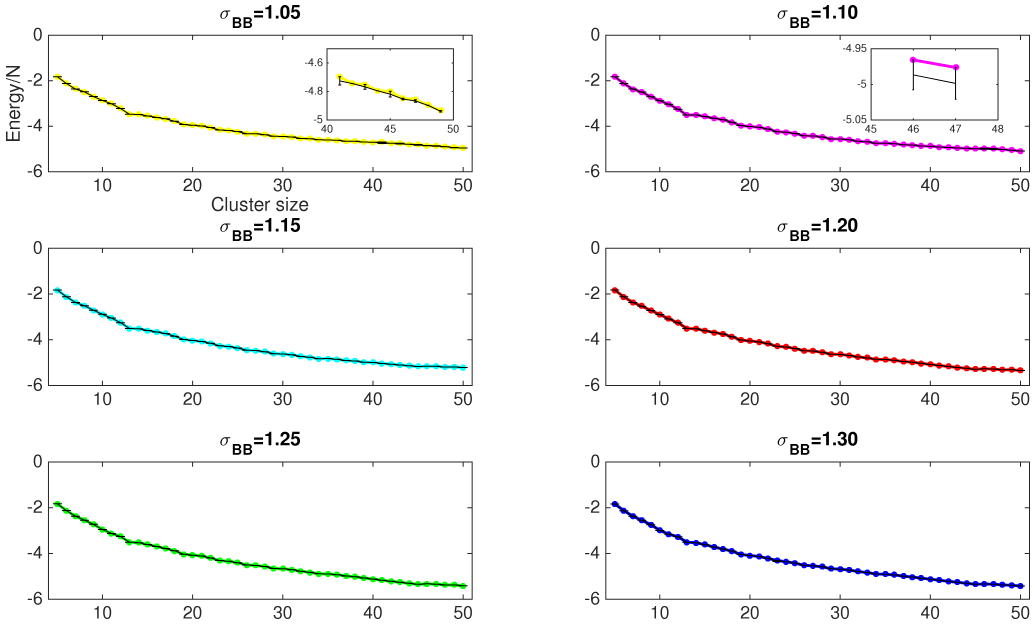


Figure 7.5: Energies generated by MIWD+PerOp for the different σ_{BB} using Knead as perturbation operator. Insets show the energy difference of clusters with suboptimal energies to energies in CCD.

Cut and Splice variant in this study (CutSpliceVar) is loosely based on a similar procedure done on LJ clusters by Deaven and co-workers [Deaven *et al.*, 1996]. To adapt to the binary case, instead of combining hemispheres from two different clusters, type A particles of one cluster are combined with type B particles of another cluster. To further increase the chance of generating new clusters, the particles are randomly displaced before the combination. This allows preservation of the compositional aspect of the cluster and no further repair is needed.

There was no expectation when this variant was developed for this study but it surprisingly generated higher success rates (Figure 7.6) compared to Kneading. Figure 7.7 shows the best energies generated by MIWD+PerOp with CutSpliceVar as perturbation operator while Figure 7.8 compares these energies to the GO found in CCD. Figures with insets show the cluster instances with which MIWD+PerOp obtained suboptimal energies.

In most of the instances where Kneading performed poorly, Cut Splice Variant performed exceedingly well. It however failed in a number of intances most notably in larger cluster instances where type A and type B atoms are almost the same size ($\sigma_{BB} = 1.05, 1.10$), an observation similar to that of Kneading.

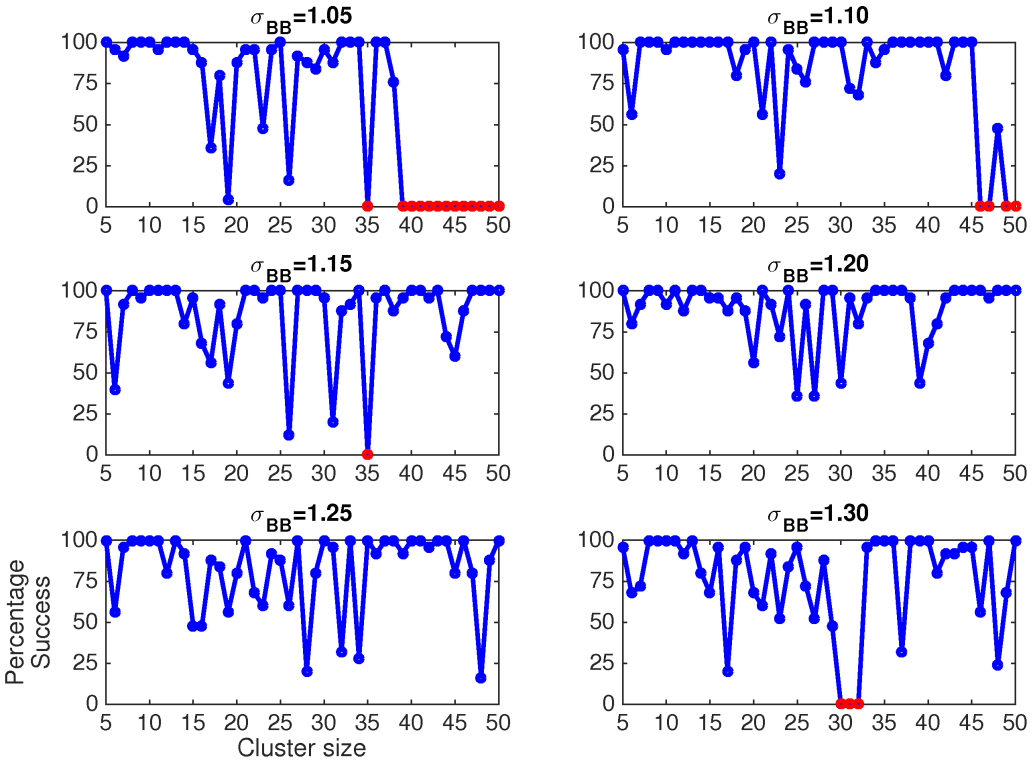


Figure 7.6: MIWD+PerOp success rates for the BLJ clusters up to size 50 using Cut Splice Variant in Phase 2. With the exception of $\sigma_{BB} = 1.05, 1.10$, there is no indication that the success rates declines for MIWD+CSVar as cluster size increases.

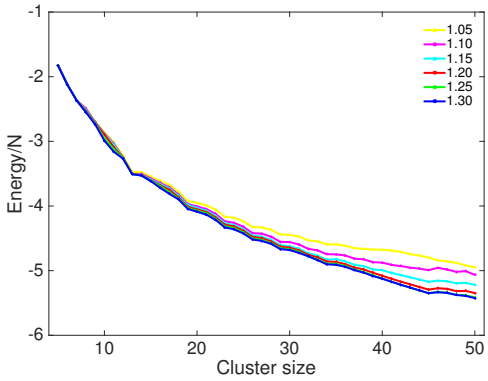


Figure 7.7: Energies generated by MIWD+PerOp for the different σ_{BB} using CutSpliceVar as perturbation operator.

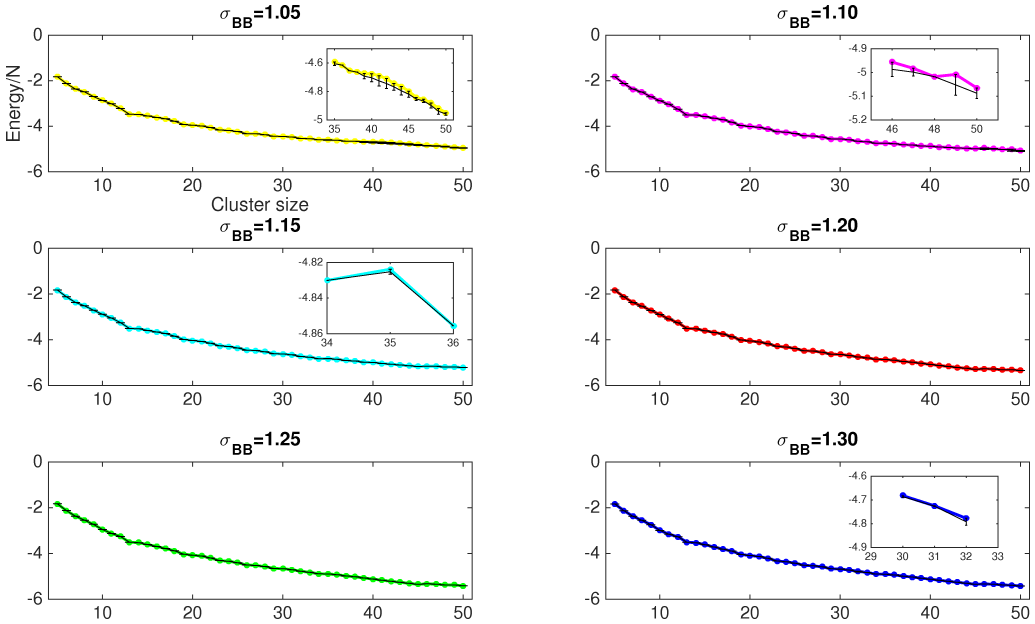


Figure 7.8: Energies generated by MIWD+PerOp for the different σ_{BB} using CutSpliceVar as perturbation operator. Insets show the energy difference of clusters with suboptimal energies to energies in CCD.

MIWD+PerOp, specifically MIWD+Knead and MIWD+CutSpliceVar, performed generally well on most cluster instances of BLJ apart from 9 instances in MIWD+Knead and 21 instances in MIWD+CutSpliceVar. This makes MIWD+Knead able to hit the putative GO 287 instances out of the 296 while 275 out of 296 instances for MIWD+CutSpliceVar. Tables 7.1 and 7.2 show the instances for which both MIWD+Knead and MIWD+CutSpliceVar failed to obtain the GO plus a few more instances where MIWD+CutSpliceVar obtained suboptimal energies. Among the failed test instances, $N = 43 - 45$ for $\sigma_{BB} = 1.05$, and $N = 47$ for $\sigma_{BB} = 1.10$ and $N = 43$ for $\sigma_{BB} = 1.30$ were also found to be difficult cluster instances in a study of Binary LJ using Evolutionary Algorithm (EA) [Marques & Pereira, 2010]. In EA, however, they found the global optimum of BLJ_{43} for $\sigma_{BB} = 1.30$ with $\sim 13\%$ success rate while MIWD+CutSpliceVar obtained a significant $\sim 92\%$.

To be able to clearly see how these configurations with suboptimal energies compare to the configurations of the putative GO in CCD, analyses on the radial packing of particles in each configurations were done. Specifically, the average radial distribution functions (rdf), $gave_{\alpha,\beta}$ (r) (Equation 7.1), of the MIWD results and the GO were calculated and compared.

$$gave_{\alpha,\beta}(r) = \frac{\sum_{h=1}^{nBins} g_{\alpha,\beta}(r_h)}{nBins} \tag{7.1}$$

where

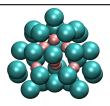
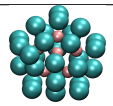
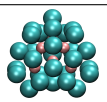
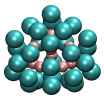
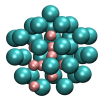
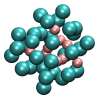
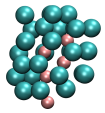
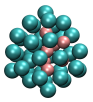
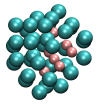
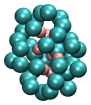
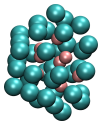
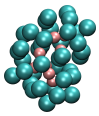
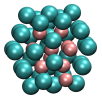
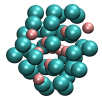
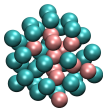
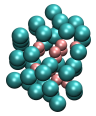
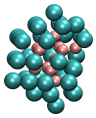
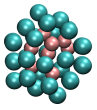
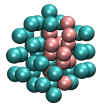
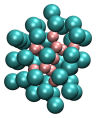
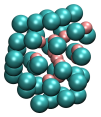
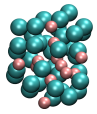
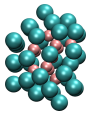
$$g_{\alpha,\beta}(r_h) = \frac{n_{\alpha,\beta}(r_h)}{4\pi r^2 \Delta r \frac{N}{V}} \tag{7.2}$$

where $nBins$ is the number of bins/shells being considered, r_h is the hth shell from a reference particle α, $g_{\alpha,\beta}(r_h)$ is the probability of finding a particle β on the $r_h th$ shell of the reference particle α, $n_{\alpha,\beta}(r_h)$ is the number of β particles on the $r_h th$ shell of the reference particle α, $4\pi r^2 \Delta r$ is the volume of the r_hth shell (with Δr as the width of the shell and r as the midpoint of shell r_h), N is the number of particles in the cluster while V is the volume of the cluster (each particle in the cluster is given 1 mass unit).

A peak in the graph means that there is a concentration of target particles β which are distance r away from the reference particle α. Three types of average rdf were calculated for the suboptimal MIWD+PerOp clusters namely: $gave_{A,A}(r)$, $gave_{B,B}(r)$, and $gave_{A,B}(r)$, where A and B refers to particle type A and B, respectively.

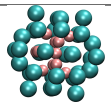
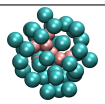
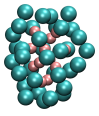
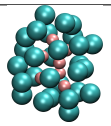
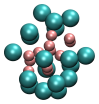
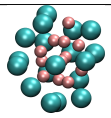
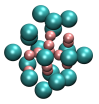
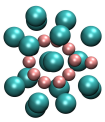
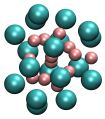
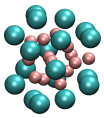
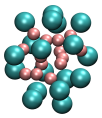
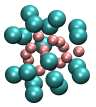
For average rdfs where at least one of the MIWD+PerOp obtained the GO (Appendix figures D.1, D.2, D.3, D.4 D.5 and D.6), both MIWD+Knead

Table 7.1: Instances for which MIWD+PerOp obtained suboptimal energies. The geometries with energies in red text are global optima.

σ_{BB}	Cluster Size	MIWD+ Knead	MIWD+ CutSpliceVar	CCD GO
1.05	35	 **-161.19033**ε	 **-160.76735**ε	 **-161.19033**ε
	40	 **-188.14674**ε	 **-187.03993**ε	 **-188.14674**ε
	41	 **-192.54037**ε	 **-192.25804**ε	 **-193.74307**ε
	43	 **-204.27769**ε	 **-204.01781**ε	 **-205.00331**ε
	45	 **-215.94559**ε	 **-215.89913**ε	 **-216.85633**
	46	 **-222.98576**ε	 **-222.71777**ε	 **-223.21059**
	47	 **-228.28117**ε	 **-228.19229**ε	 **-228.72554**ε
	48	 **-235.01686**ε	 **-234.23414**ε	 **-235.07922**ε
	49	 **-241.74579**ε	 **-241.00815**ε	 **-242.01160**ε

and MIWD+CutSpliceVar results with suboptimal energies closely matched the number of peaks of the GO. There is an exception for BLJ$_{40}, \sigma_{BB} = 1.05$ under MIWD+CutSpliceVar (Appendix figures D.4f, D.5f and D.6f) where there are lower concentrations of type A to type A and type B to type B particles in the

Table 7.2: Instances for which MIWD+PerOp obtained suboptimal energies. The geometries with energies in red text are global optima.**CCD does not have the configuration file for GO of BLJ$_{46}$, $\sigma_{BB} = 1.10$. Authors were contacted but has not responded.

σ_{BB}	Cluster Size	MIWD+ Knead	MIWD+ CutSpliceVar	CCD GO
1.10	46	 **-229.12496**ε	 **-226.98879**ε	**-226.74585**ε xyz file unavailable**
	47	 **-234.17312**ε	 **-234.17312**ε	 **-234.93427**ε
1.15	35	 **-168.88256**ε	 **-168.83265**ε	 **-168.88256**ε
1.30	30	 **-140.58202**ε	 **-140.42822**ε	 **-140.58202**ε
	31	 **-146.51693**ε	 **-146.46602**ε	 **-146.51693**ε
	32	 **-153.34842**ε	 **-152.87604**ε	 **-153.34842**ε

first shell or peak [1] compared to the GO. This resulted into having more type A particles closer into the surface of the cluster than the GO.

For instances where none of the MIWD+PerOp obtained the GO, Appendix figures D.7 and D.8 show $gave_{AA}(r)$ for MIWD+Knead and MIWD+CutSpliceVar, respectively. These figures show that there are less type A particles gathered around type A particles in the first shell compared to the putative GO. This behaviour resulted in a domino effect where MIWD+PerOp results were not able to match the succeeding shells of the putative GO. This lower concentration of type A particles in the first shell also resulted into MIWD+PerOp clusters generating more shells than the putative global optima after the first shell which is apparent around distance $r = 1.6$ in most of the plots. This pushes

[1] A shell or peak in this context can be defined as a concentration of target particles distance r away from a reference particle.

type A particles nearer to the surface which can be verified from the ball and stick representation of the MIWD+PerOp resulting clusters (Tables 7.1 and 7.2). The only exception to this trend is instance BLJ_{47} for $\sigma_{BB} = 1.10$ which shows better geometric conformity to the putative GO.

Appendix figures D.9 and D.10, on the other hand, show $gave_{BB}(r)$ for MIWD+Knead and MIWD+CutSpliceVar, respectively. This time MIWD+PerOp have more type B particles gathered in the first shell from a type B reference particle compared to the putative GO. Shells thereafter of MIWD+PerOp could not properly match that of putative GO except again for the instance BLJ_{47} for $\sigma_{BB} = 1.10$. This behaviour is expected as a result of lesser type A particles gathered around the first shell pushing more number of type B to occupy that shell.

Lastly, appendix figures D.11 and D.12 show $gave_{AB}(r)$ for MIWD+CutSpliceVar and MIWD+Knead, respectively. The peak for the first shell between type A to type B particles appears to have better conformity compared to $gave_{AA}(r)$ and $gave_{BB}(r)$ for most of MIWD+Knead clusters although succeeding shells differ greatly from the putative GO. Surprisingly, better conformity from the second shell onwards can be seen not only for BLJ_{47} for $\sigma_{BB} = 1.10$ both for MIWD+Knead and MIWD+CutSpliceVar but for BLJ_{48} for $\sigma_{BB} = 1.05$ under MIWD+Knead (Appendix figure D.12f) and BLJ_{49} for $\sigma_{BB} = 1.05$ under MIWD+Knead (Appendix figure D.12g), as well.

7.2.3 Combination of Phase 2 Operators

The percentage success rates showed that MIWD+CutSpliceVar can generally beat MIWD+Knead in most of the test instances but it was the latter that can effectively find the correct basin at which the lowest energy lies. On the other hand, for the test intances where MIWD+Knead and MIWd+CutSpliceVar generated suboptimal energies, the calculated average rdf showed that geometries have more type A particles inching closer to the surface which in turn moved some of the type B particles towards the centre. Additionally, there were a few instances where MIWD+Knead showed good conformity to the putative GO under the $gave_{A,B}(r)$. These results prompted for a different kind of approach to Phase 2 of the algorithm. Taking advantage of MIWD+CutSpliceVar's high success rate of rediscovering the putative GO as well as MIWD+Knead's more robust performance in hitting the putative GO in more test instances and the knowledge that suboptimal clusters can be made to conform with the putative GO by pushing a particle type (type A in the cases presented above) into the centre, the author decided on implementing a combination of perturbation operators as part of Phase 2 of the algorithm. This means that operators would be applied

alternatively to the clusters and iterated several times with the aim of generating better and optimal cluster energies. It was reasonable to include into the combination the energy-based swapping operators, EBS_HALB and EBS_HBLA , to combine with Knead or Cutsplice variant since the average rdf of the suboptimal clusters showed that proper "shelling" could be achieve by swapping particle types which is the very idea of these operators. The following combination of operators were implemented as Phase 2 of the algorithm: (1) CutSpliceVar + Knead (CSKnead), (2) CutSpliceVar + EBS_HALB (CSHALB), (3) CutSpliceVar + EBS_HBLA (CSHBLA), (4) Knead + EBS_HALB (KneadHALB), and (5) Knead + EBS_HALB (KneadHBLA).

Additionally diversification of population members was included together with the combination of operators to allow for wider exploration of the potential energy surface. This is accomplished by a measure called D_{cut} which is calculated using (Equation 7.3) :

$$D_{cut} = \omega D_{ave} \tag{7.3}$$

$$D_{ave} = \frac{\sum_{i=1}^{N_C} \sum_{j=i+1}^{N_C-1} d(X_i, X_j)}{C_2^{N_C}} \tag{7.4}$$

where

$$d(X,Y) = \sum_{j=1}^{Natoms} (OrdX_j - OrdY_j)^2 + 2\left|N_{TA}^X - N_{TA}^Y\right| \tag{7.5}$$

where ω is the degree of dissimilarity from the average and D_{ave} (Equation 7.4) is the average dissimilarity of the current population, $C_2^{N_C}$ is the combination operator which gives the total number of possible pairings of clusters in the population, X and Y are two distint cluster configurations, $Natoms$ is the total number of particles in the cluster, $OrdX_j$ ($OrdY_j$) is a vector of size $Natoms$ containing the Euclidean distance between each particle and the geometric centre of cluster X (Y) ordered in a non-increasing way while N_{TA}^X (N_{TA}^Y) is the number of particles of type A in cluster X (Y). Since all clusters in the population have the same number of type A and B particles, the term, $2\left|N_{TA}^X - N_{TA}^Y\right|$, is always zero in this case.

The D_{cut} measure is based on an optimization method applied to the binary LJ system by Cassioli and co-workers [Cassioli *et al.*, 2009]. Before a low energy cluster is accepted into the population, it has to be a certain D_{cut} (Equation 7.3) distance away or dissimilar from the parent cluster(s). This replacement

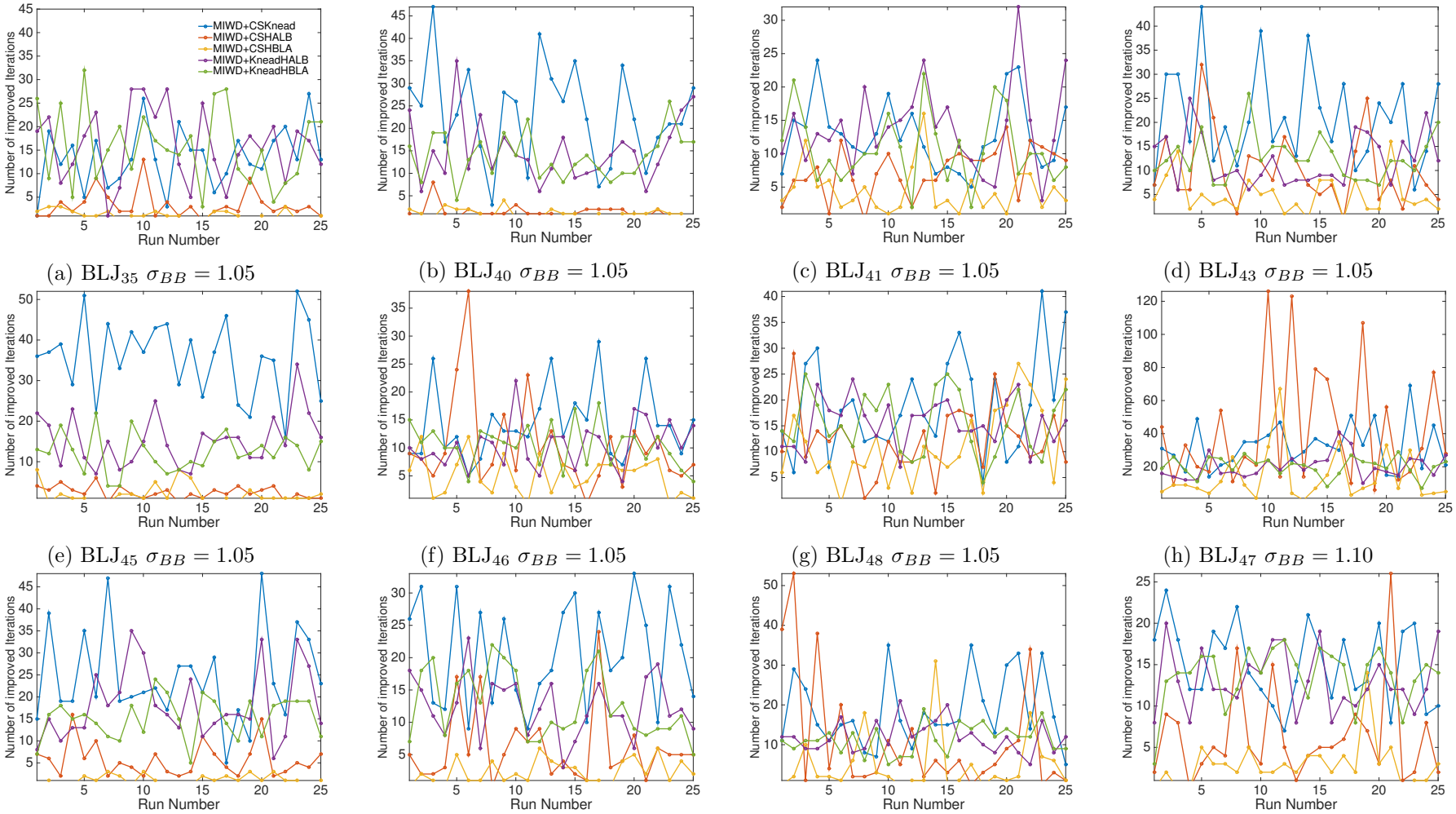
strategy is described as "not too greedy" as we are guaranteed that the population is monotonically not worsening but at the same time not too quick to accept new clusters with only a small bit of energy decrease. Table 7.3 shows the indicators whether each combination achieved the GO or not. MIWD+CombiOp successfully obtained the GO for some of the BLJ instances where MIWD+PerOp failed. The worst performing of the 5 combinations is MIWD+CSHBLA where it was not able to hit the GO of any of the test instances. MIWD+KneadHALB hit 6 of the 15 instances while the rest of the MIWD+CombiOp only obtained either 3 or 4 out of 15 instances. The instances for which MIWD+PerOp failed proved to be very difficult ones even for the combination of operators as well. This is not surprising as an algorithm for BLJ called EA [Marques & Pereira, 2010] for binary atomic clusters claimed that cluster instances $N = 43 - 45$ for $\sigma_{BB} = 1.05$, $N = 47$ for $\sigma_{BB} = 1.10$ and $N = 43$ for $\sigma_{BB} = 1.30$ were the most difficult instances generating very low success rates in their experiments.
where

Table 7.3: Performance of MIWD+CombiOp at the BLJ instances with suboptimal results. **Y** indicates success to arrive at the GO while **N** indicates otherwise.

	MIWD+CombiOp				
Size,σ_{BB}	CSKnead	CSHALB	CSHBLA	KneadHALB	KneadHBLA
35,1.05	**Y**	N	N	**Y**	**Y**
40,1.05	N	N	N	**Y**	N
41,1.05	N	N	N	N	N
43,1.05	N	N	N	N	N
45,1.05	N	N	N	N	N
46,1.05	N	N	N	N	N
47,1.05	N	**Y**	N	N	N
48,1.05	N	N	N	N	N
49,1.05	N	N	N	N	N
46,1.10	N	N	N	N	N
47,1.10	N	N	N	**Y**	N
35,1.15	**Y**	**Y**	N	**Y**	**Y**
30,1.30	**Y**	N	N	N	N
31,1.30	N	**Y**	N	**Y**	N
32,1.30	**Y**	N	N	**Y**	**Y**

The number of improved energies from the start to the last iteration in each run for MIWD+CombiOp were recorded for the purpose of determining which combination of operators are able to explore the potential energy surface better. Plots in Figure 7.9 show these values for some of the difficult test instances done for 25 different runs. The lines connecting the values are just guides for the eyes. There is an obvious poor performance from MIWD+CSHALB and

MIWD+CSHBLA with most of the plots found near the bottom of the rest with only a few spikes observed in Figures 7.9f, 7.9h, 7.9k and 7.9l. MIWD+CSKnead, among all MIWD+CombiOp, shows a capability of exploring more low energies over most of the runs while MIWD+KneadHALB and MIWD+KneadHBLA are average performers. These results however tell us that obtaining more improved (lower) energies does not necessarily follow generating the lowest energy configurations at the end of each run. One evidence is in Figure 7.9h where both MIWD+CSKnead and MIWD+CSHALB generated more improved energies on most runs but it was MIWD+KneadHALB that was able to hit the GO (see Table 7.3) for BLJ_{47} under $\sigma_{BB} = 1.10$.



(a) BLJ_{35} $\sigma_{BB} = 1.05$ (b) BLJ_{40} $\sigma_{BB} = 1.05$ (c) BLJ_{41} $\sigma_{BB} = 1.05$ (d) BLJ_{43} $\sigma_{BB} = 1.05$

(e) BLJ_{45} $\sigma_{BB} = 1.05$ (f) BLJ_{46} $\sigma_{BB} = 1.05$ (g) BLJ_{48} $\sigma_{BB} = 1.05$ (h) BLJ_{47} $\sigma_{BB} = 1.10$

(i) BLJ_{35} $\sigma_{BB} = 1.15$ (j) BLJ_{30} $\sigma_{BB} = 1.30$ (k) BLJ_{31} $\sigma_{BB} = 1.30$ (l) BLJ_{32} $\sigma_{BB} = 1.30$

Figure 7.9: Comparison of improvements over iterations of the different MIWD+CombiOp for some of the difficult BLJ test instances. MIWD+CSHALB and MIWD+CSHBLA show least improvement over iterations while MIWD+CSKnead appears to find more lower energies as iterations progress in most of the instances.

7.3 Summary and Conclusion

Modified IWD together various perturbation operators were successfully applied to most of the test instances under the binary LJ system for up to size 50 for 6 values of σ_{BB}. Four perturbation operators were used in Phase 2 of MIWD+PerOp. Three of which are based on previously published works, modified in this study and applied for the first time to binary LJ system. The energy-based swapping perturbation operators, EBS_HALB and EBS_HBLA, performed poorly even on low cluster sizes and thus were not tested on cluster sizes beyond 20. MIWD+Knead was able to hit more GO out of the 296 BLJ test instances compared to MIWD+CutSpliceVar but it was the latter which performed consistently high in terms of success rates. Geometries of the test instances for which MIWD+PerOp failed to hit the GO were analyzed using comparisons of the average rdfs with that of the GO. It was found out that most of the configurations with suboptimal energies had more type A particles pushed into the surface effecting some type B particles to be positioned more into the inner core.

A combination of perturbation operators for Phase 2 of the algorithm were then implemented with the aim to address the problem and a diversification of population step was added to limit acceptance of new low energy clusters to dissimilar clusters. Five combinations of perturbation operators (MIWD+CombiOp) were implemented namely MIWD+CSKnead, MIWD+CSHALB, MIWD+CSHBLA, MIWD+KneadHALB and MIWD+KneadHBLA. Runs of MIWD+CombiOp on the test instances where MIWD+PerOp failed proved to be difficult still although MIWD+KneadHALB was able to hit the GO for 6 out of the 15 instances. Number of improved energies were compared for the different MIWD+CombiOp which generally showed the ability of MIWD+CSKnead to find more lower energies in one run although it was not as successfull in finding the GO at the end of the runs than MIWD+KneadHALB. The energy-based swap operators (HALB and HBLA) as single operators in MIWD+PerOp did not show promise in finding the GO even on small Binary LJ clusters but they were effective auxiliary operators in MIWD+CombiOp specifically when combined with the Knead pertubation operator. MIWD+PerOp and MIWD+CombiOp together were able to successfully rediscover most of the GO for the Binary LJ system up to size 50 for six different values of σ_{BB}. Despite combining two perturbation operators for Phase 2 of the algorithm, GO of some of the test instances proved to be difficult to find. There were no further attempts to modify the current implementation or add more iterations as this proved to have less gain.

Chapter 8

MIWD+PerOp for Morse Cluster Optimization

8.1 Introduction

The Morse potential is another model that has been used to test robustness of optimization algorithms due to the presence of different structural geometries compared to the icosahedral-dominated LJ global optima (GO) geometries. Despite only depending on the separation between pairs of atom, the potential energy surface becomes more rugged with decreased range [Doye & Wales, 1996]. Earlier studies also stated that minimization using random initial configurations are much less likely to find the GO for short-ranged Morse potentials due to the larger number of local minima [Bytheway & Kepert, 1992;Stillinger & Stillinger, 1990]. Furthermore, barrier heights are likely to be higher making it more difficult to overcome when solutions are stuck in local minima. The parameter α in Equation 1.11 defines the range of interaction of particles in the Morse potential. At $\alpha = 6$, the Morse potential has the same curvature at the bottom of the well as the LJ potential. For this value, the structural behaviour, generally icosahedral, of the two systems is very similar [Doye *et al.*, 2004]. However, the potential energy landscape of Morse clusters changes characteristic with α. As α is increased from $\alpha = 6$ to $\alpha = 14$, the number of minima increases rapidly turning the landscape from icosahedral to decahedral to close-packed clusters. Hence this is an ideal system for configurational problems providing tougher criteria than LJ clusters. In the succeeding sections, MIWD is tested for $\alpha = 6$ to further validate its performance on locating generally icosahedral structures and for $\alpha = 14$ to determine its generality on locating different structural geometries.

8.2 Results

MIWD with a Phase 2 perturbation operator, Grow–and–Etch (MIWD+PerOp), has been applied to Morse clusters from size $N = 5$ to 60. All results from MIWD+PerOp were compared with the known putative GO listed in the Cambridge Cluster Database (CCD). Success rates in hitting the known putative GO (out of 25 runs) is shown in Figure 8.1. MIWD+GrowEtch was able to successfully rediscover the energies of all the test instances for $\alpha = 6.0$ and mostly all of the test instances for $\alpha = 14.0$ missing only 5 instances (M_{47}, M_{55}, M_{57}, M_{58} and M_{60}). There is a clear indication of difficulty for the algorithm to hit the GO beyond N=25 under the short-range potential $\alpha = 14.0$ and beyond $N = 50$ under the long-range potential $\alpha = 6.0$. A study on finding GO for Morse clusters called *TP-MSBH* [Doye *et al.*, 2004] found out that Morse clusters from $N = 41 - 47$ at $\alpha = 14.0$ are indeed difficult cases due to the very flat decahedral shapes motif of the GO. Runs of their algorithm also generated zero success rates, when forward/backward procedure [1] are not counted, for test instances $N = 42, 44, 45, 54, 55, 61, 80$ under $\alpha = 14.0$. Increased number of random runs, without forward or backward procedure, allowed *TP-MSBH* to find the GO configurations for these sizes which was reported to be never more than 1000 runs. In MIWD+PerOp, Morse cluster sizes $N = 42, 44, 45$, and 54 were found without needing forward or backward optimization procedure or increasing the number of runs. The *TP-MSBH* algorithm, although largely succesful over sizes $N = 41 - 80$ under $\alpha = 14.0$, used two-phase local searches where one of the phases transformed the Morse potential function to bias between spherical and oblate or prolate structures. The parameters that are needed to be tuned in the transformed Morse potential in *TP-MSBH* rendered the method less transferable to other systems.

With MIWD+GrowEtch successfully applied to the generally icosahedral LJ clusters, it was also expected to find the lowest energy geometries for the long-range potential $\alpha = 6.0$ where structures generally follow polytetrahedral, icosahedral geometries. Overlayed configurations for selected test instances generated by MIWD+GrowEtch under the long-range potential $\alpha = 6.0$ are shown in Figures 8.2a, 8.2b and 8.2c. These three lowest-energy geometries have LJ_{55} (or M_{55}) as the base structure. M_{58} is composed of M_{55} plus 3 particles forming a triangle attaching itself on one of the icosahedral faces. M_{59}, on the other hand, is an M_{55} plus 4 particles forming a square attaching itself over one of the icosahedral vertices. Finally, M_{60} is the M_{59} geometry with the remaining particle positioning itself between the 4 particles forming a square and on of the

[1] Takes an optimal cluster with one less or one more particle and optimizing it by placing or taking an atom on the surface of the cluster.

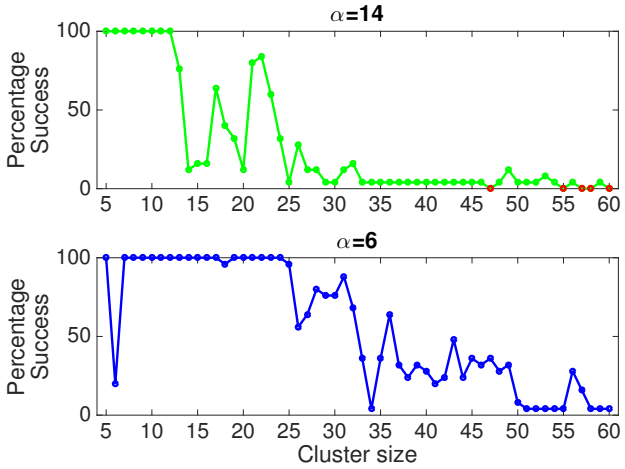
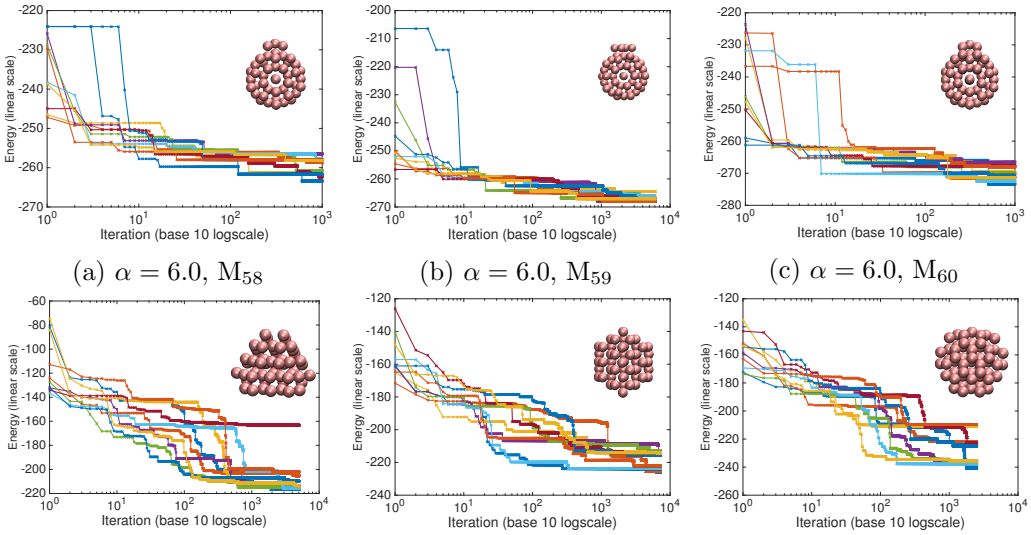


Figure 8.1: MIWD+GrowEtch success rates for Morse clusters up to size $N = 60$. Markers in red indicate failure to find the putative GO.

icosahedral vertices.

It would be worth comparing the results of this algorithm to the performance of RSA (random search algorithm) as presented in a study comparing it to Birmingham GA [Johnston, 2003]. RSA is a simple random search algorithm that generates a random initial population and minimizes the energy of these structures using L-BFGS. RSA results for M_{20}, M_{30}, M_{40} and M_{50} under $\alpha = 6.0$ showed that it only successfully found the GO for the lowest cluster size tested M_{20}. MIWD+GrowEtch, on the other hand, was able to find the GO for all the mentioned test instances with relatively high success hit rates most specifically for M_{20} (100%), M_{30} (76%) and M_{40} (28%) (Figure 8.1). This tells us that a purely random search algorithm, even with an effective energy minimization step, is insufficient to locate the proper basin of the global minimum.

The search trajectory of MIWD+GrowEtch for selected test instances are shown in Figure 8.2. For clarity, especially during the first few iterations, the plots are presented in a semi-logarithmic scale where the iteration number axis is set to base-10 logarithmic scale while the energy axis is set to linear scale. In all cases there is a rapid improvement in the energies (a sharp drop within the first 10 iterations) in the early iterations. There is a pronounced difference in the progress of low energy search between the $\alpha = 6.0$ (subfigures 8.2a, 8.2b, and 8.2c) and $\alpha = 14.0$ plots (subfigures 8.2d, 8.2e, and 8.2f). For $\alpha = 6.0$, the algorithm is able to jump quite steeply near the basin that leads to the icosahedral GO while for 14.0, traversal paths of IWD agents show hopping to several local minima before finding the correct basin. This difference provides more evidence of a larger number of minima for $\alpha = 14.0$ GO structures than in $\alpha = 6.0$. Furthermore, this also shows the ability of MIWD+GrowEtch to traverse the Morse potential



(a) $\alpha = 6.0$, M_{58} (b) $\alpha = 6.0$, M_{59} (c) $\alpha = 6.0$, M_{60}

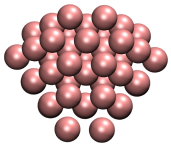
(d) $\alpha = 14.0$, M_{54} (e) $\alpha = 14.0$, M_{56} (f) $\alpha = 14.0$, M_{59}

Figure 8.2: Search trajectories of 10 IWD agents generated by MIWD+GrowEtch for selected Morse cluster sizes for both $\alpha = 6.0$ and 14.0. Overlayed configuration is final configuration of trajectory of the best IWD agent with energy matching that of the GO in CCD.

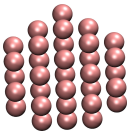
surface between configurations with small energy differences.

Figure 8.3 shows the structures of test instances where MIWD+GrowEtch failed to hit the GO. Even in these suboptimal structures, the geometry follows the fcc-like (Figure 8.4) packing expected of Morse clusters under the short-range potential parameter $\alpha = 14.0$. The GO configurations for these test instances are not published in the CCD and so the difference in actual structure cannot be compared. A comparison of the difference in energies of the obtained geometries with the published energies in CCD are presented in Table 8.1 instead.

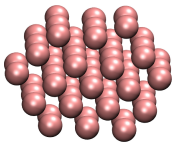
Furthermore, in the absence of the GO configurations for $\alpha = 14.0$ with which we can compare the structures of the suboptimal results in this study, we took the GO configurations generated in this study having $N-1$ and $N+1$ particles and compare it with the suboptimal M_N. In Figure 8.5 MIWD+GrowEtch GO configurations for M_{54}, M_{56} and suboptimal M_{55} are projected onto a plane such that greatest number of particles overlap. The three configurations are exactly similar in the stacking of particles from the topmost layer with the bottom layer of the suboptimal M_{55} varying quite clearly from both M_{54} and M_{56}. Furthermore, both the projected M_{54} and M_{56} have a mirror plane symmetry, as shown by the (red) vertical line which the suboptimal M_{55} does not have. From these geometry comparisons, it can be conjectured that GO configuration for M_{55} under $\alpha = 14.0$ may be an M_{54} with the remaining particle attaching itself to the bottom-most layer, or simply an M_{56} with one particle (the one with the fewest



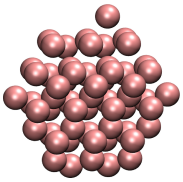
(a) M_{47}
$V(r) = -183.4113120115D_e$



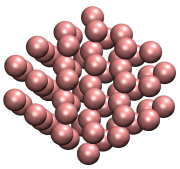
(b) M_{55}
$V(r) = -219.5251565405D_e$



(c) M_{57}
$V(r) - 229.5487319415D_e$



(d) M_{58}
$V(r) - 234.6714731752D_e$



(e) M_{60}
$V(r) - 244.5720168073D_e$

Figure 8.3: Configurations of the clusters with best (suboptimal) energies, $V(r)$, generated by MIWD+GrowEtch with their associated energies under the short-range potential $\alpha = 14.0$.

nearest neighbours) taken out of the bottom layer where the plane of symmetry passes through. This conjectured GO M_{55} geometry was generated from M_{56}, as stated above, and its calculated its energy ($V = -220.6462D_e$) found to be similar to the energy published in the CCD.

The same procedure was attempted for M_{47}; however looking along the planes of symmetry for the GO configurations of M_{46} and M_{48} (Figure 8.6) shows that there is no growth pattern from size 46 to 48. Conjecturing the GO configuration of M_{47}, which could either have M_{46}- or M_{48}- based geometry, is not straightforward. Furthermore, the suboptimal configuration generated from MIWD+GrowEtch does not share geometry with either the optimal M_{46} and M_{48}. It is, however, likely that the optimal M_{47} would follow from M_{46} by completing a C_s symmetry with the addition of one particle near the bottom layer. $M48$ already has a complete C_s symmetry and taking out the most bottom particle to preserve its symeetry would cause a strain on the surrounding particles, generating possibly higher energies than generating M_{47} from M_{46}. This example proves to show that these Morse clusters can have very different geometries even for neighbouring sized instances making it all the more challenging for stochastic optimization algorithms to find the GO. This analysis was only added to show whether a forward or backward optimization could be done from available MIWD+GrowEtch GO results but is not an actual part of the algorithm in this study.

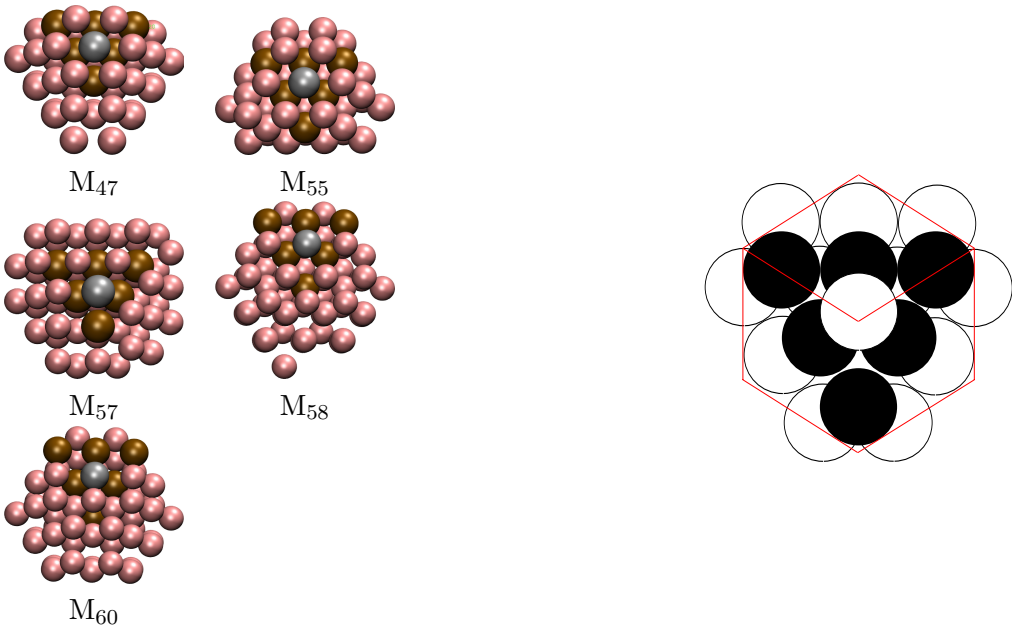
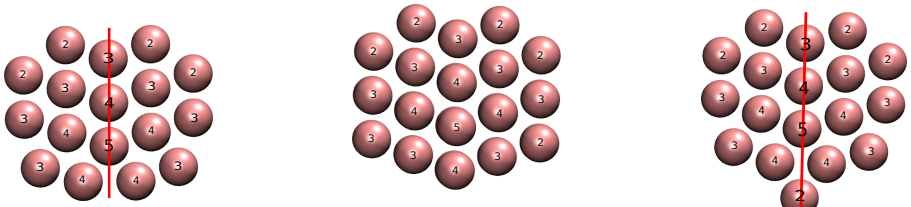


Figure 8.4: Suboptimal geometries generated by MIWD+GrowEtch for $\alpha = 14.0$ highlighting the particles making up the fcc-like structure of the geometries. Figure on the right shows the schematic diagram of a complete fcc-packing of spheres.

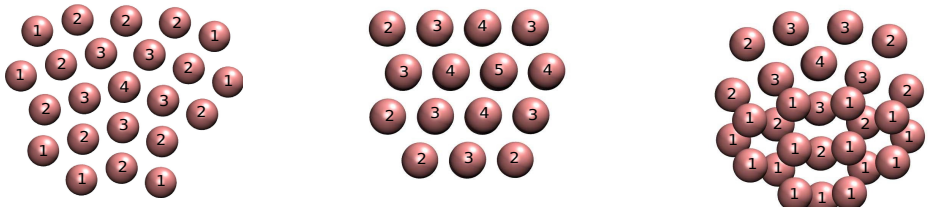


(a) M_{54} GO

(b) M_{55} MIWD+GrowEtch Suboptimal

(c) M_{56} GO

Figure 8.5: M_{54}, M_{55} and M_{56} configurations generated by MIWD+GrowEtch projected onto a plane. Values on the particles indicate the number of particles on that projection.



(a) M_{46} GO

(b) M_{47} MIWD+GrowEtch Suboptimal

(c) M_{48} GO

Figure 8.6: M_{46}, M_{47} and M_{48} configurations generated by MIWD+GrowEtch projected onto a plane. There is no obvious growth pattern from the GO of M_{46} to M_{48} to deduce the structure of the GO for M_{47}.

Table 8.1: Difference in energies of Morse in CCD and MIWD+GrowEtch results for M_{47}, M_{55}, M_{57}, M_{58}, and M_{60} under $\alpha = 14.0$.

Size	MIWD+GrowEtch	CCD	Energy Difference
47	-183.411312ε	-183.508227ε	0.096915ε
55	-219.525157ε	-220.646208ε	1.121051ε
57	-229.548732ε	-230.663986ε	1.115254ε
58	-234.671473ε	-234.809078ε	0.137605ε
60	-244.572017ε	-244.579066ε	0.007049ε

8.3 Summary and Conclusion

Modified IWD together with GrowEtch (MIWD+GrowEtch) was attemped to find the global optima of clusters containing up to 60 atoms bound by the Morse potential as a function of the range of the interatomic force. Two interactomic forces with different characteristic length scales, generated using $\alpha = 6.$ and 14.0, were tested. MIWD+GrowEtch was able to rediscover all test instances for $\alpha = 6.0$ and all but 5 test instances with $\alpha = 14.0$. The 5 instances for which MIWD+GrowEtch failed to find the GO were known to be some of the most difficult clusters in a study of Morse cluster using a two-phase basin-hopping (TP-MSBH) alogirhtm with Morse potential transformation. In TP-MSBH, these difficult instances where discovered with either a forward or backward procedure, or by increasing the number of random runs together with tests on various parameter settings associated with the transformed Morse potential.

The suboptimal structures generated by MIWD+GrowEtch were found to be following the fcc-like geometries expected of optimal Morse clusters under the short-range potential $\alpha = 14.0$. Energy trends of runs of the two range potentials show that the potential energy surface for $\alpha = 14.0$ indeed contains larger number of local minima as evidenced by the larger number of downsteps. This also indicates the ability of MIWD+GrowEtch's effective exploitation cability in finding lower energies.

All in all, MIWD+GrowEtch has been shown to successfully rediscover a large percentage (95%) of the 112 Morse GO with little parameter tuning needed compared to the best algorithm in literature.

Chapter 9

MIWD+CombiOp for Janus Cluster Optimization

9.1 Introduction

Janus clusters, whose stability depends both on angular and spatial interaction of particles, could well be the most complex in all of the systems tested in this study. In this study, it appears that the homonuclear LJ and Morse clusters are the most trivial systems for geometry optimization while the Binary LJ system places the complexity a step further with the additional compositional factor of each particle aside from its spatial variable. It is easy to see that Janus cluster optimization is more complex than optimizing Binary LJ clusters due to the fact that both spatial and angular variables in Janus clusters involve continuous variables as opposed to the compositional variable in Binary LJ clusters which is discrete.

As far as this study is involved, Janus clusters have not yet been studied for optimality of cluster geometry up to size 50 and size 100 using a nature-inspired algorithm and using a potential energy based on the LJ potential function with the attraction term modulated by the angular interaction of two-patched Janus particles. The model that was used to describe interaction between two Janus particles in this study is defined in Equation 9.1 (details in Section 2.4).

$$V_{Janus}\left(r_{i,j},\theta_{ij},\theta_{ji}\right)=\begin{cases}\sum\limits_{i=1}^{n-1}\sum\limits_{j>i}^{n}V_{LJ}\left(r_{i,j}\right) & r_{i,j}<\sigma_{i,j}\\ \sum\limits_{i=1}^{n-1}\sum\limits_{j>i}^{n}V_{LJ}\left(r_{i,j}\right)MV_{angle}\left(\hat{\boldsymbol{r}}_{\boldsymbol{i,j}},\theta_{ij},\theta_{ji}\right) & r_{i,j}\geq\sigma_{i,j}\end{cases} \tag{9.1}$$

where

$$MV_{angle}\left(\hat{\boldsymbol{r}}_{\boldsymbol{i,j}},\theta_{ij},\theta_{ji}\right)=f\left(\theta_{ij}\right)f\left(\theta_{ji}\right) \tag{9.2}$$

$$f(\theta) = -exp\left(-\frac{\theta^2}{2\sigma^2}\right) + exp\left(-\frac{(\theta - 180)^2}{2\sigma^2}\right) \tag{9.3}$$

The significant factor in the model is the angular term that modulates the attraction between two Janus particles. The angular term in this study was defined for a two-patched Janus particle where patch A (B) is only attractive to patch A (B) of another Janus particle. The parameter σ in the angular term (Equation 2.23) proved to be instrumental in generating a desirable geometry as will be discussed in the succeeding sections below. The following sections show results obtained from MIWD+CombiOp consisting of modified IWD for Phase 1 and a combination of Janus cluster appropriate perturbation operators for Phase 2. The geometries obtained have the lowest energies based on runs of the MIWD algorithm and as far as published literature.

9.2 Results

Four perturbation operators were used for Phase 2 of the Janus system, namely the standard rotation/orientation (*Or*) and displacement (*Dis*) of particles, Growing and Etching (*GrowEtch*) of particles originally used for the LJ system and a localized version of particle rotation called Neighbor Optimization (*Neigh*). Using a single operator to find the next possible low energy configuration by simply displacing the particles would be less beneficial for this system as the orientation of each particle with its neighbors is a crucial part of the geometry structure, thus MIWD+CombiOp was implemented combining one spatial and one orientational perturbation operator. Monte Carlo simulations of Janus clusters found in recent literature have also implemented spatial and orientational moves in each particle to generate new configurations [Giacometti *et al.*, 2012;Sciortino *et al.*, 2010;Fantoni *et al.*, 2014].

This study tested 6 different combinations of the operators, details of which are in Section 2.7.4, to further optimize the clusters generated by Phase 1 of the algorithm. These are referred to here as MIWD+OrDis, MIWD+NeighDis, MIWD+GrowEtchOr, MIWD+OrGrowEtch, MIWD+GrowEtchNeigh and MIWD+NeighGrowEtch. Structures of geometries associated with the lowest energies obtained in this study were also analysed in one of the following sections. As there is no known optimal configurations or putative GO for the test instances tested for Janus clusters, the runs are terminated by a maximum number of iterations or until the energy difference in succeeding iterations are sufficiently close to each other.

9.2.1 Angular Term-dependent geometry

The angular term in the model used to describe the orientation interaction of Janus particles showed to be a crucial part in this study. The parameter σ showed that it dictates the geometry of the lowest energy structures. Two values for $\sigma = \{\frac{180^\circ}{2}, \frac{180^\circ}{6}\}$ were tested for all MIWD+CombiOp to show this. Tables 9.1 and 9.2 show the lowest energies up to size 15 obtained for the different MIWD+CombiOp results for $\sigma = \frac{180^\circ}{6}$ and $\sigma = \frac{180^\circ}{2}$, respectively.

Without exception, all MIWD+CombiOp results associated with $\sigma = \frac{180^\circ}{6}$ resulted in chain-like configurations (Table 9.1). To recall from the orientation interaction contour maps (Figure 2.7) defining $\sigma = \frac{180^\circ}{6}$, particles at perfect alignment with each other have stronger attraction than when $\sigma = \frac{180^\circ}{2}$ thus allowing particles to attach more strongly to a hemisphere with no particle bound to it yet rather than attach itself to aggregates where energy contribution is possibly higher caused by a more "angled" or disaligned interaction. It is mentioned in a study of numerical simulations of off-balance amphiphilic Janus particles by Hong and co-workers [Hong & Cacciuto, 2012] that larger attractive surfaces imply that particles have more flexibility to rotate about their axes resulting into an expanded space of structural configurations. It can also be observed in Table 9.1 across all MIWD+CombiOp results, specially for cluster sizes 3 and 4, that configurations are similar but with slightly different energies. These differences can be accounted from orientational interaction differences between each particle which are not easily visible from the ball-and-stick images. The chain-like configurations generated in this study from setting $\sigma = \frac{180^\circ}{6}$ however do not show promising geometries and so further simulations under this σ value was discontinued.

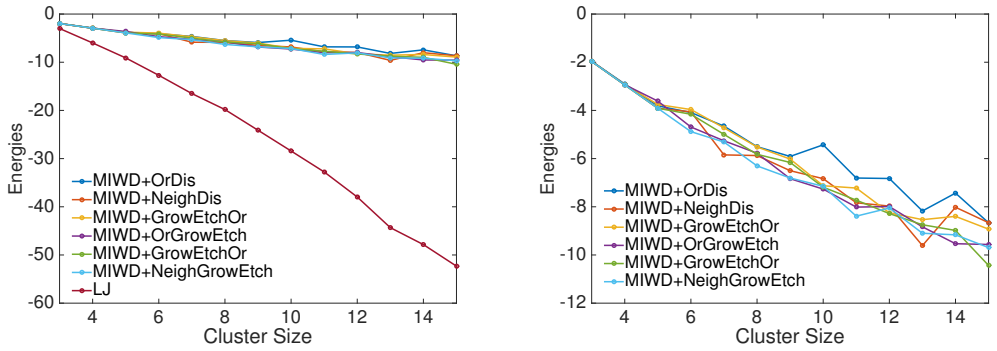


Figure 9.1: Lowest energies of structures generated by MIWD+CombiOp when $\sigma = \frac{180^\circ}{6}$ for clusters up to size 15.

More compact and structured configurations were observed when $\sigma = \frac{180^\circ}{2}$ across most combinations of perturbations operators with exception from MIWD+OrDis and MIWD+NeighDis. Table 9.2 shows the different geometries

associated with the lowest energies obtained for the different MIWD+CombiOp up to size 15. For each cluster size in Table 9.2, geometries with the lowest energies are highlighted in red. It appears that a combination of Grow and Etch and Orientation mutation found the geometries with the lowest energies with the exception of J_3 which was obtained from the MIWD+NeighDis results. Figure 9.2 shows the plots of lowest energies for all MIWD+CombiOp results up to size 15 showing linearly decreasing energies as cluster size increases with MIWD+GrowEtchOr generally obtaining the lowest energies. MIWD+GrowEtchOr was further tested on larger cluster sizes up to size 50 and Figure 9.3 shows the plot of the lowest energies obtained.

Grow and Etch, as applied in Janus clusters, adds a number of Janus particles to a random position in the surface of the cluster and etches high-energy particles one by one until cluster size is back to its original number while Orientation mutation randomly rotates the orientation of each Janus particle. Grow and Etch proves to be very efficient in traversing the potential energy surface for lower energy structures as this is also the best performing operator under LJ and Morse systems. The promising lowest energies obtained from MIWD+GrowEtchOr using $\sigma = \frac{180^\circ}{2}$ was the deciding factor for which more runs were done on larger Janus cluster sizes under this combination. Table 9.3 shows the structures generated for Janus clusters up to size 50 and size 100 while Figure 9.4 shows the energy search trajectories as iterations progress under MIWD+GrowEtchOr. The ability of the algorithm to find lower energy configurations is shown by these various plots showing an exponential decay of energies for all cluster agents. Inspecting the contribution of each operator into the energy decrease, GrowEtch tended to provide relatively significant energy jumps while the Orientation Mutation operator provides more minimal but effective energy decrease. The latter acted as a repair operator for the weaker orientational interaction of particles caused by the change in spatial arrangement of particles.

The best configurations obtained from MIWD+GrowEtchOr were analysed based on : (1) basic building blocks observed on these structures, (2) the average orientation each particle makes within its neighbourhood, and (3) compactness of the geometries. These are all presented in subsection 9.2.2.

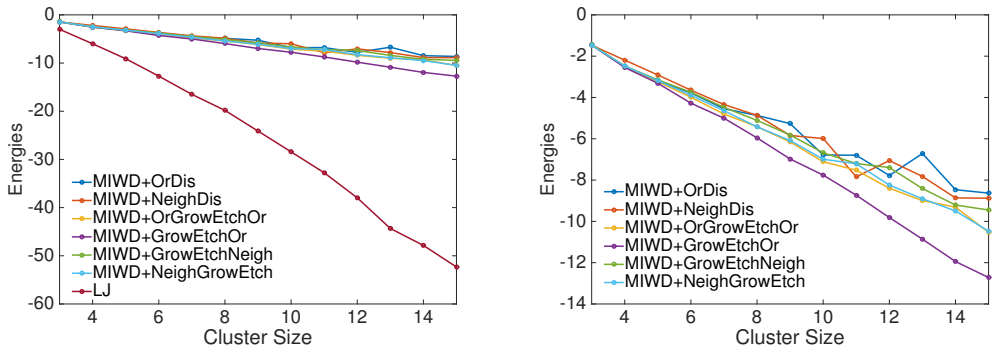


Figure 9.2: Lowest energies of structures generated by MIWD+CombiOp when $\sigma = \frac{180^\circ}{2}$ for clusters up to size 15.

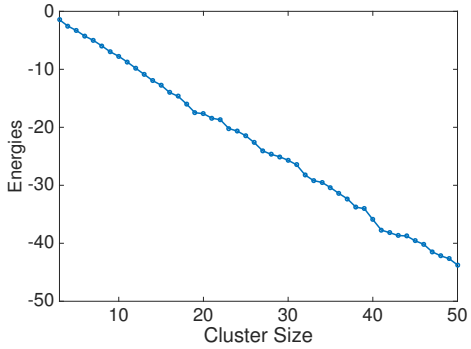
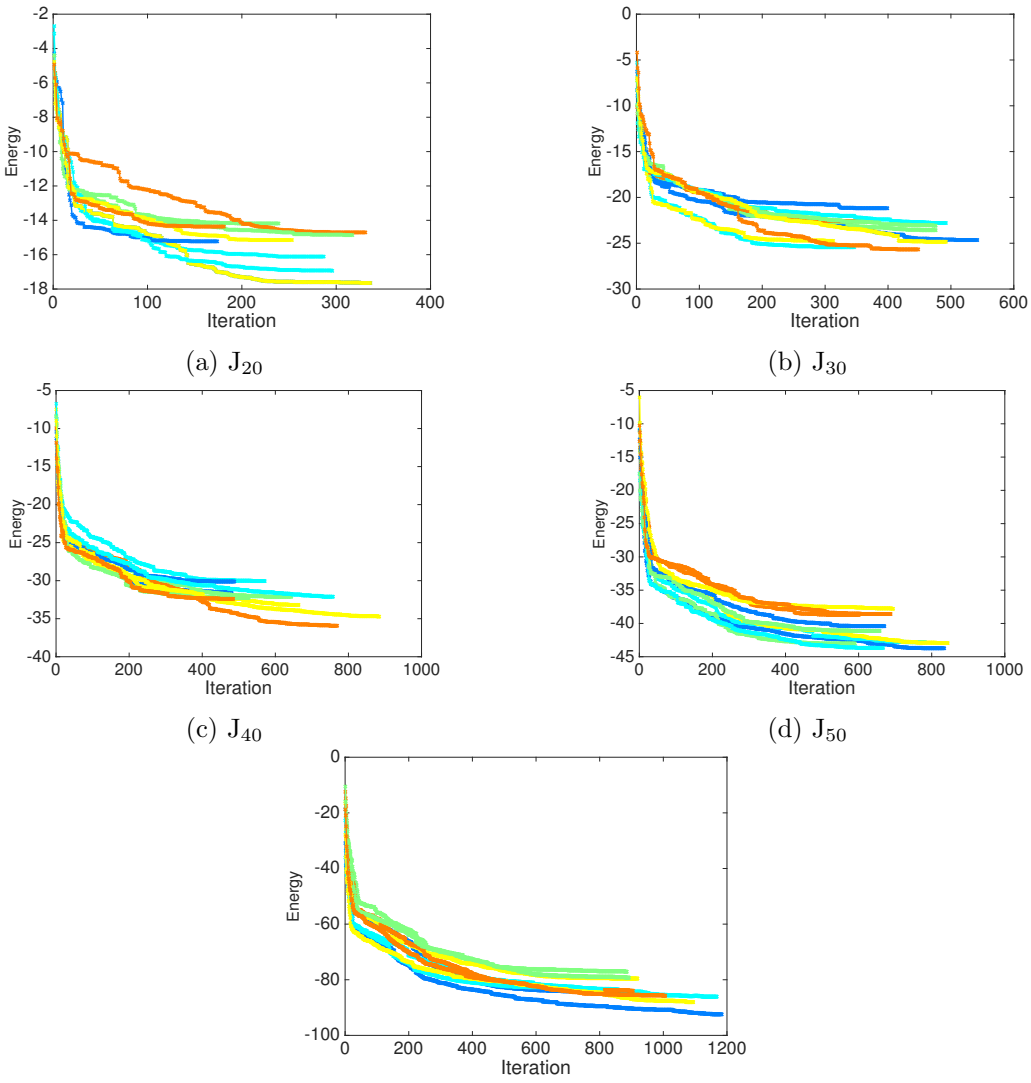


Figure 9.3: Best energies obtained for Janus clusters up to size 50 with MIWD+GrowEtchOr (with the exception of J_3 which was obtained using MIWD+NeighDis).



(a) J_{20}

(b) J_{30}

(c) J_{40}

(d) J_{50}

(e) J_{100}

Figure 9.4: Search trajectories of 10 IWD agents, differentiated by colour, generated by MIWD+GrowEtchOr when $\sigma = \frac{180^{\circ}}{2}$ for Janus cluster sizes 20, 30, 40, 50 and 100. IWD agents with shorter plots indicate no energy change in the succeeding iterations.

Table 9.1: Geometries with lowest energies generated for different MIWD+CombiOP combinations under $\sigma = \frac{180}{6}°$ for $3 \leq N \leq 15$.

Size	OrDis	NeighDis	GrowEcthOr	OrGrowEcth	GrowEtchNeigh	NeighGrowEtch
3	 -1.958947ε	 -1.968531ε	 -1.965295ε	 -1.964037ε	 -1.969083ε	 -1.968940ε
4	 -2.902636ε	 -2.940454ε	 -2.932488ε	 -2.925947ε	 -2.940701ε	 -2.940854ε
5	 -3.811078ε	 -3.910817ε	 -3.613544ε	 -3.754124ε	 -3.910447ε	 -3.911329ε
6	 -4.088540ε	 -4.058922ε	 -4.688418ε	 -3.963454ε	 -4.159253ε	 -4.881108ε
7	 -4.651010ε	 -5.846411ε	 -5.261597ε	 -4.722336ε	 -4.994779ε	 -5.313122ε
8	 -5.504048ε	 -5.873473ε	 -5.776206ε	 -5.516476ε	 -5.822943ε	 -6.302251ε
9	 -5.910194ε	 -6.501374ε	 -6.834102ε	 -6.018749ε	-6.159284ε	-6.805212ε
10	-5.428327ε	-6.833016ε	-7.265462ε	-7.126687ε	-7.187097ε	-7.149494ε

Continued on next page

Table 9.1 – continued from previous page

Size	OrDis	NeighDis	GrowEcthOr	OrGrowEcth	GrowEtchNeigh	NeighGrowEtch
11	 -6.806617ε	 -7.833132ε	 -8.012681ε	 -7.222689ε	 -7.724715ε	 -8.389009ε
12	 -6.825823ε	 -7.971460ε	 -7.991369ε	 -8.251125ε	 -8.282879ε	 -8.050877ε
13	 -8.173572ε	 -9.609777ε	 -8.832686ε	 -8.534496ε	 -8.745703ε	 -9.098163ε
14	 -7.431837ε	 -8.019493ε	 -9.532715ε	 -8.397965ε	 -8.981718ε	 -9.165060ε
15	 -8.664762ε	 -8.660139ε	 -9.564911ε	 -8.928361ε	 -10.434055ε	 -9.693502ε

Table 9.2: Geometries with lowest energies generated for different MIWD+CombiOP combinations under $\sigma = \frac{180^\circ}{2}$ for $3 \leq N \leq 15$. Energies in red are lowest energies found.

Size	OrDis	NeighDis	GrowEcthOr	OrGrowEcth	GrowEtchNeigh	NeighGrowEtch
3	 -1.456909ε	 -1.470657ε	 -1.455783ε	 -1.456657ε	 -1.448431ε	 -1.469547ε
4	 -2.521895ε	 -2.192473ε	 -2.548211ε	 -2.492816ε	 -2.477071ε	 -2.476890ε
5	 -3.196579ε	 -2.909816ε	 -3.312383ε	 -3.235568ε	 -3.143495ε	 -3.184901ε
6	 -3.778017ε	 -3.645365ε	 -4.275585ε	 -3.987443ε	 -3.750446ε	 -3.895399ε
7	 -4.544014ε	 -4.341701ε	 -5.006151ε	 -4.802059ε	 -4.479187ε	 -4.642378ε
8	 -4.869105ε	 -4.866642ε	 -5.957121ε	 -5.416227ε	 -5.108715ε	 -5.420130ε
9	 -5.265110ε	 -5.839715ε	 -6.980199ε	 -6.153247ε	-5.819609ε	-6.086337ε
10	-6.783709ε	-5.989870ε	-7.758908ε	-5.839715ε	-6.678985ε	-6.988927ε

Continued on next page

Table 9.2 – continued from previous page

Size	OrDis	NeighDis	GrowEcthOr	OrGrowEcth	GrowEtchNeigh	NeighGrowEtch
11	 -6.798454ε	 -7.841632ε	 -8.734168ε	 -7.522213ε	 -7.199464ε	 -7.212099ε
12	 -7.777852ε	 -7.056764ε	 -9.813512ε	 -8.397292ε	 -7.392476ε	 -8.242759ε
13	 -6.713804ε	 -7.824209ε	 -10.863190ε	 -8.995341ε	 -8.409720ε	 -8.905366ε
14	 -8.466292ε	 -8.863198ε	 -11.944162ε	 -9.312378ε	 -9.209163ε	 -9.496877ε
15	 -8.632359ε	 -8.878281ε	 -12.720319ε	 -10.528153ε	 -9.449232ε	 -10.474012ε

Table 9.3: Geometries with lowest energies generated from MIWD+GrowEtchOr with $\sigma = \frac{180^\circ}{2}$ for $16 \leq N \leq 50$ and $N = 100$.

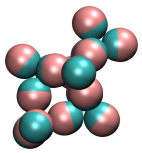
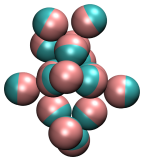
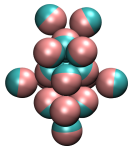
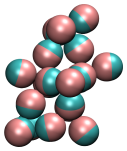
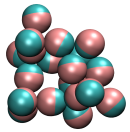
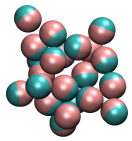
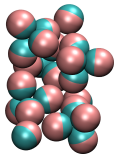
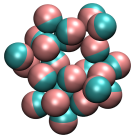
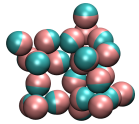
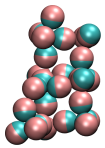
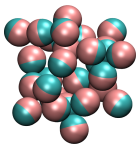
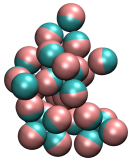
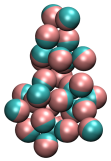
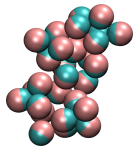
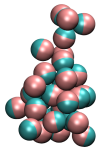
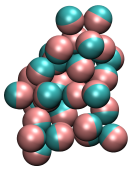
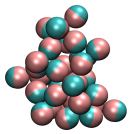
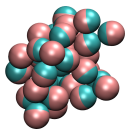
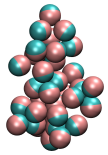
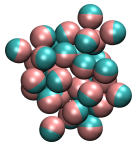
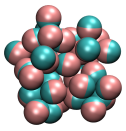
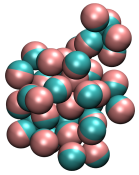
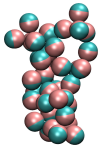
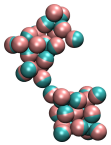
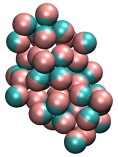
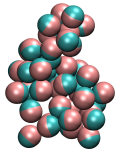
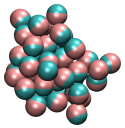
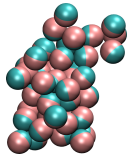
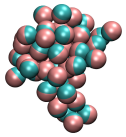
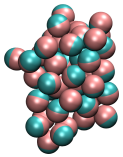
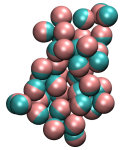
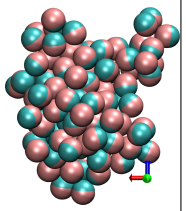
MIWD+GrowEtchOr				
16  -13.991847ε	17  -14.636913ε	18  -15.975920ε	19  -17.484214ε	20  -17.643858ε
21  -17.565174ε	22  -18.700470ε	23  -20.210991ε	24  -20.656301ε	25  -21.463641ε
26  -22.654789ε	27  -24.056137ε	28  -24.659036ε	29  -25.085689ε	30  -25.684634ε
Continued on next page				

Table 9.3 – continued from previous page

MIWD+GrowEtchOr				
31  -26.464229ε	32  -28.215359ε	33  -29.190757ε	34  -29.499188ε	35  -30.392609ε
36  -31.409113ε	37  -32.340730ε	38  -33.771384ε	39  -34.057033ε	40  -35.924451ε
41  -37.720348ε	42  -38.322123ε	43  -38.661668ε	44  -40.142873ε	45  -40.229610ε

Continued on next page

Table 9.3 – continued from previous page

MIWD+GrowEtchOr				
46  -40.387619ε	47  -41.465621ε	48  -42.149504ε	49  -42.648980ε	50  -43.980277ε
100				
100  -92.631001ε				

9.2.2 Janus cluster geometrical properties

The orientation a particle makes within its neighbouring particles could dictate the geometry it would make as a cluster. Perfect orientation with another particle, given a sufficient distance of separation betwee them, would render lower energy contribution between them but higher energy for another particle wanting to attach itself to the pair. A desired cluster is expected to have a good combination of highly orientation particles and more compact geometry that would result into the lowest energy possible. Figure 9.5a (blue plot) shows the average orientation measure that each particle makes within its neighbouring particles (within $1.2\sigma_{LJ}$ radius from center of the particle). The orientation measure is calculated from $MVang$ (Equation 2.23) where perfect alignment of similar patches under $\sigma = \frac{180^{\circ}}{2}$ would have a pair potential of $\sim 0.80V_{LJ}(r_{ij})$ energy units and $\sim -0.80V_{LJ}(r_{ij})$ otherwise. The more positive the orientation measure is the more the particles are closer to perfect alignment with its neighbouring particles. As J_3's lowest energy geometry resulted in a linear arrangement of particles (Table 9.2, row 1, column NeighDis), it is thus expected to have the most positive orientation measure in all test instances. More compact geometries for J_3 (Table 9.2, row 1, columns *GrowEtchOr* and *GrowEtchNeigh*) have higher energies than a chain of three Janus particles. This was the only exception as comparison of lowest energies for $4 \geq N \geq 15$ between chain-link geometries (Table 9.2 column *NeighDis*) and more globular geometries (Table 9.2 column *GrowEtchOr*) showed that the latter had lower more desirable energies.

Average orientation measures across different Janus cluster above size 3 ranges between 0.20 to 0.45. Translating it to angles, on the average, particles make an angle of between 34° to 51° around its neighbourhood (Figure 9.5a right y-axis). There is no clear indication that particles are more disaligned as cluster size increases due to measures oscillating within the range even from neighbouring cluster sizes. There is, however, an indication that the geometries are able to maintain a neighbourhood of connectivity with a limited range of angular interactions suggesting a well, if not highly, ordered structure. These are investigated and discussed in a later paragraph.

Compactness of the geometries were calculated based on the same measure, ρ^2 (Equation 6.1), computed for LJ clusters. Figure 9.5b provides a view of the compactness of the best energies generated for Janus clusters up to size 50 in comparison with the LJ GO as a reference. It is expected that compactness of clusters from size 3 to 50 will have an increasing trend owing to the fact that the next larger cluster has one more particle but the oscillating behaviour from one cluster size to the next tells us that there is no obvious geometrical pattern of growth for the Janus clusters generated in this study. More so after J_{25} where

compactness values have more pronounced peaks and gradual deviation from LJ compactness measure suggesting an evidence of divergence from the common icosahedral cluster motif.

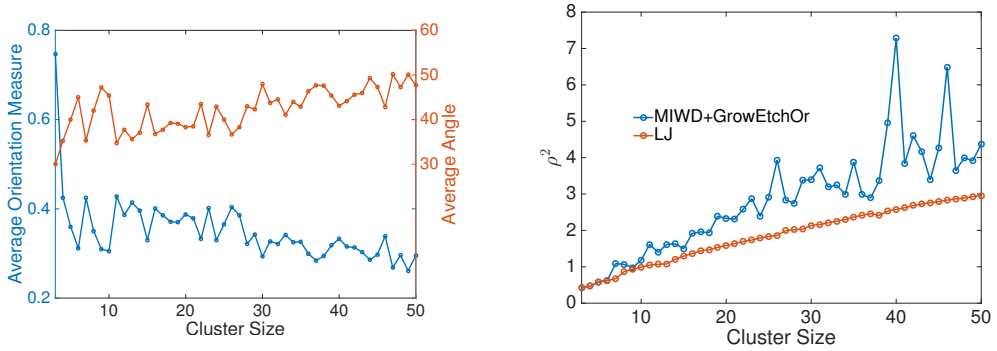


Figure 9.5: (a) Average orientation measure each Janus particle makes with its neighbouring particles calculated for each MIWD+CombiOp Janus cluster results.(b) Comparison of compactness of MIWD+CombiOp Janus cluster results with the CCD global optima LJ clusters calculated using the squarred hyperradius measure ρ^2.

Despite the geometries having no clear evidence of growth structure, structures from J_4 to J_{15} appears to have an inherent common smallest building block. The structures were observed to contain a basic three-particle neighbourhood with each particle making a $\sim 60^\circ$ angle with the other two neighbours (Figure 9.6). This smallest building block was used to characterize the surrounding neighbourhood. In each generated geometry, all three-particle neighbourhoods were identified and the plane containing them were recorded. All neighbouring particles within $1.2\sigma_{LJ}$ from a position slightly above the center of the plane (the vector defining this position from the plane is orthogonal to the plane) were counted. Neighbourhood from either side of the plane are counted separately. Calculating these for all the generated structures with lowest energies up to size 50 generated 6 more common building blocks. A three-particle neighbourhood with no particle count on one side of its plane indicates that the 3-particle neighbourhood is part of a surface. The 6 other building blocks were 4-,5-,6-,7-,8- and 9-nearest neighborhood blocks shown in ball and stick representation in Figure 9.7. The plots in the figure is inclusive of the triplet of particles that defined the reference plane. A 4-nearest neighbor block indicates a tetrahedral or J_4 structure, 5-nearest neighbors indicates skewed trigonal bipyramid, 6-nearest neighbors indicates presence of J_6 structure or rectangular bipyramid, 7-nearest neighbors indicates pentagonal bipyramid , 8-nearest neighbors indicate a slightly-skewed or irregular antiprism closed by two parallelograms, and finally 9-nearest neighbors indicate another

irregular antiprism closed by a pentagon and a parallelogram.

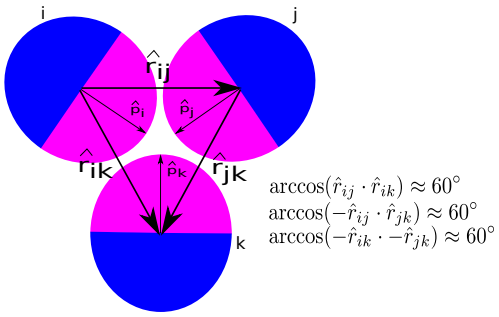


Figure 9.6: The smallest basic building block, a triangle of particles, observed in generated Janus clusters. The interparticle vector on each pair of particles makes an approximately 60° angle with each other.

All these building blocks are present in the geometry generated for J_{38} and Figure 9.8 shows the distribution of these building blocks within the cluster. The building blocks that dominate the structure are skewed trigonal bipyramids and rectangular bipyramids. Calculating the distribution of building blocks for all cluster instances, Figure 9.9 shows that there is a shift of building block composition from triplets of particles and tetrahedral blocks (Figure 9.9a) for cluster sizes 10 to 30 to trigonal and rectangular bipyramids (Figure 9.9b) for cluster sizes 31 to 50. The decrease of occurence of the 3 particle block in the range 31 to 50 also indicates that there is an increase in volume to surface ratio. Distribution of building blocks were also calculated for the lowest energy geometry of J_{100} and both trigonal and rectangular bipyramids dominate the structure as well (Figure 9.12).

Compactness of geometry as iterations progress was also investigated for the lowest energy geometry obtained under J_{100} (Figure 9.11). These snapshots were selected from 1278 low energies generated during a run of MIWD+GrowEtchOr. Despite the iterations in this study not allowing for a precise definition of time, it is still interesting how the sequence of MIWD+GrowEtchOr configurations find lower energies as "algorithmic" time progresses. In the sequence of snapshots of configurations c to f energies are linearly decreasing but cluster is becoming less compact. Looking closely at the ball and stick representation of these configurations (Figure 9.10), the cluster appears to be extending or stretching along the z-direction (vertical blue axis in the figures) before a few particles start to find more suitable bonding sites which consequently allowed the resulting cluster to be more compact (snapshots f to m). This extending and contracting of the cluster as iterations progress is numerically presented in Figure 9.11 where the plot shows the cluster initially extending from snapshot a until snapshot f and starts to gradually contracts from

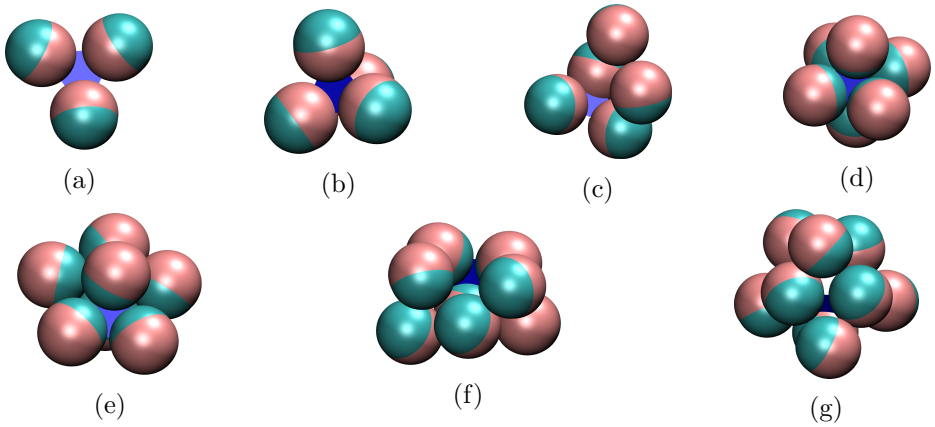


Figure 9.7: The seven building blocks observed in geometries with lowest energies generated for Janus clusters up to size 50 under MIWD+GrowEtchOr. The blue plane connecting 3 particles is the reference plane. The building blocks observed were : (a) triangle of particles, (b) tetrahedral, (c) skewed trigonal bipyramid, (d) rectangular bipyramid, (e) pentagonal bipyramid, (f) irregular antiprism closed by two parallelograms and (g) irregular antiprism closed by a pentagon and a parallelogram.

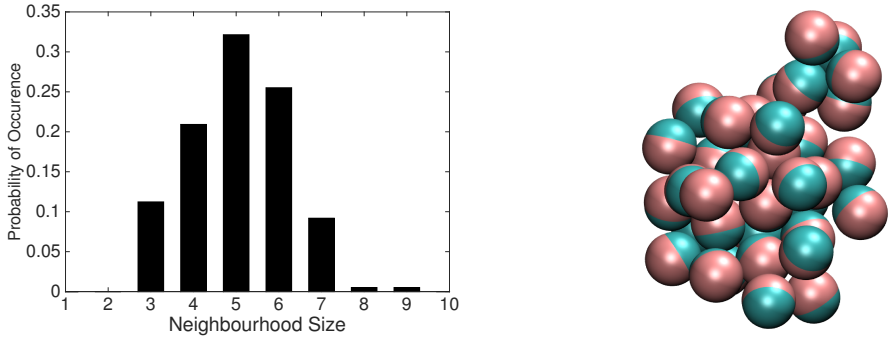
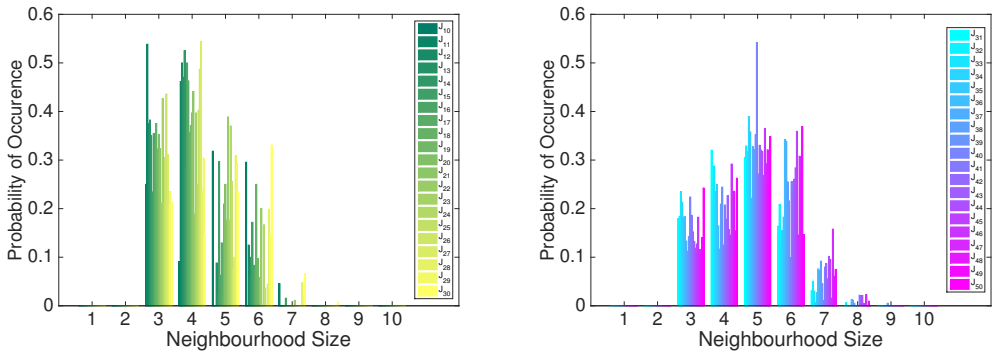


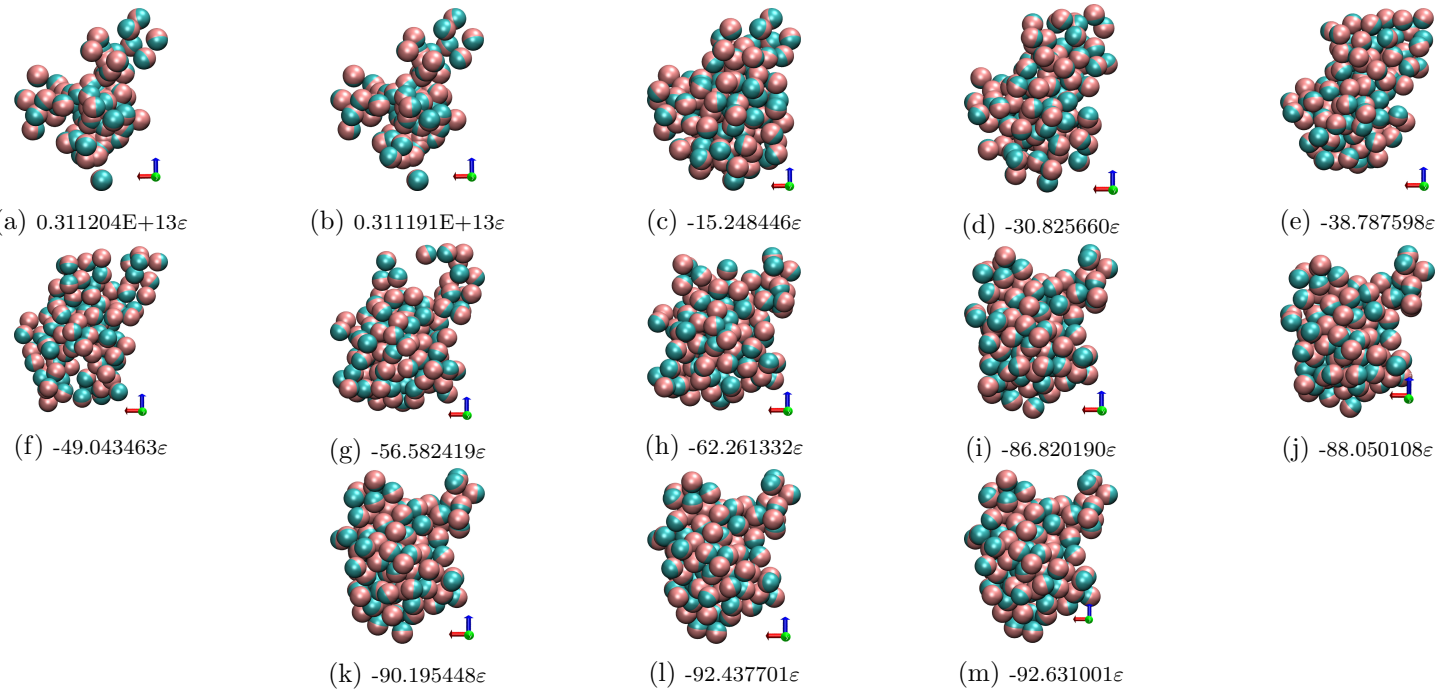
Figure 9.8: (Left) Distribution of building blocks in J_{38}. All observed building blocks are present in J_{38}. (Right) Ball-and-stick model of the structure generated by MIWD+GrowEtch.

snapshot g to snapshot m. Average orientation measure for particles within the neighbourhood of the J_{100} structure with the lowest energy was calculated to be 0.2659074 ($\sim 49.968°$) which is within the range of orientation measures observed from sizes 4 to 50.



(a) (b)

Figure 9.9: Distribution of building blocks for (a) J_{10} to J_{30} and (b) J_{31} to J_{50}. There is shift of preference from triangle to tetrahedral blocks for J_{10} to J_{30} to trigonal and rectangular bipyramids for J_{31} to J_{50}.

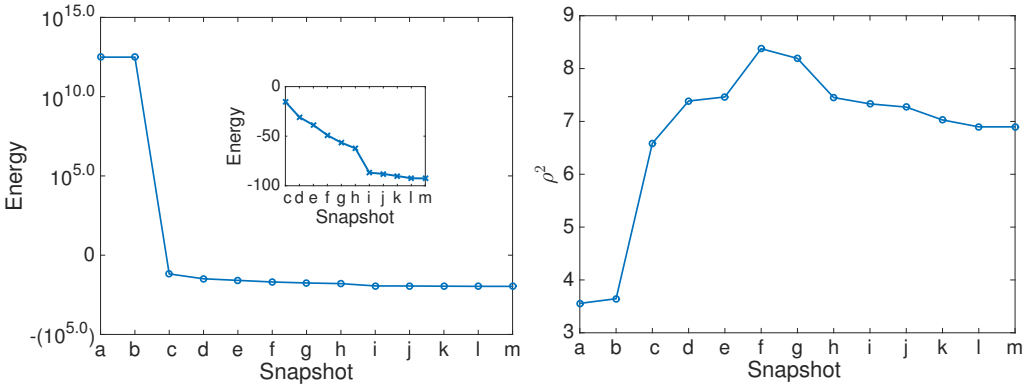


(a) 0.311204E+13ε (b) 0.311191E+13ε (c) -15.248446ε (d) -30.825660ε (e) -38.787598ε

(f) -49.043463ε (g) -56.582419ε (h) -62.261332ε (i) -86.820190ε (j) -88.050108ε

(k) -90.195448ε (l) -92.437701ε (m) -92.631001ε

Figure 9.10: Snapshots from iterations of MIWD+GrowEtchOr for J_{100}. Geometries in a and b are from Phase 1 while structures obtained from d to m are Phase 2 results. Geometry in c is the relaxed configuration resulting from Phase 1.



(a) (b)

Figure 9.11: Left : Snapshots of energies for J_{100}. Energy decrease is apparent as iteration progresses taking note of significant energy difference from starting cluster (a) to final cluster (m). Right : Compactness of clusters as iterations progress from snapshot a to m calculated using the squarred hyperradius measure ρ^2.

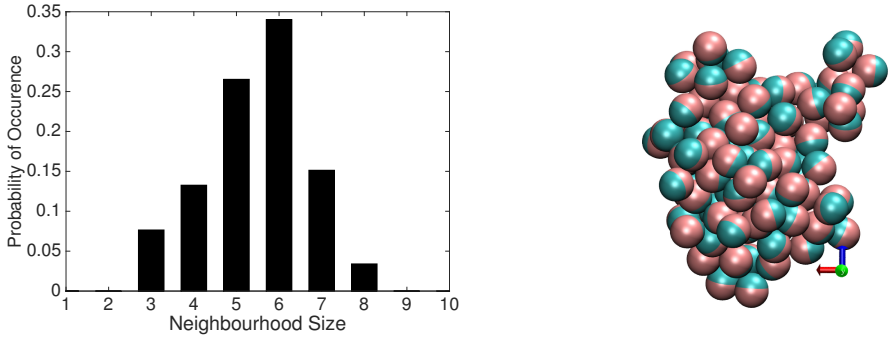


Figure 9.12: Distribution of building blocks in J_{100}. The structure is dominated by trigonal and rectangular bipyramids.

9.3 Summary and Conclusion

In this study, the Janus interaction has been modeled using an LJ potential function with the attraction term modulated by the orientational interaction of two-patched particles. The term MV_{ang} used to described the intensity of orientation between particles was tested using MIWD+CombiOp for Janus clusters up to size 50 and size 100. The lone parameter σ in MV_{ang} showed how it dictates the geometries through comparisons of geometries generated from the different MIWD+CombiOp, namely MIWD+OrDis, MIWD+NeighDis, MIWD+GrowEtchOr, MIWD+GrowEtchNeigh and MIWD+NeighGrowEtch. At $\sigma = \frac{180^\circ}{6}$, MIWD+CombiOp generated chain-like or fibrous clusters while at $\sigma = \frac{180^\circ}{2}$, MIWD+CombiOp generally generated more globe-like and struc-

tured clusters. MIWD+GrowEtchOr, an MIWD for Phase 1 and a combination of Grow–and–Etch and particle rotation for Phase 2, generated lowest energies among all other MIWD+CombiOp. Search trajectories of energies show exponential decay trend in finding lower energies. The lowest energies generated under MIWD+GrowEtchOr up to size 50 and size 100 did not show evidence of growth sequence from one cluster size to the next higher one however well-structured geometries were generated from the observation that across cluster sizes average orientation measure of each particle within its neighbourhood is within a limited range. Consequently these generated seven different building blocks identified as : triplet of particles forming $\sim 60^{\circ}$ angle with each other, 4-particle tetrahedral block, skewed 5-particle trigonal bipyramid, 6-particle rectangular bipyramid, 7-particle pentagonal bipyramid, 8-particle irregular antiprism closed by two parallelograms and 9-particle irregular antiprism closed by a pentagon and a parallelogram. Distribution of these identified building blocks were calculated across all Janus test instances and found a shift of likelihood of occurence from mostly 3- and 4-particle neighbourhodd (sizes 10-30) to mostly 5- and 6-particle neighbourhood (sizes 31-50). Furthermore, an observation of the sequence of geometries with decreasing energies for J_{100} showed that the cluster geometry volume initially expands or stretches before it starts to be more compact.

Finally, results above show that MIWD+GrowEtchOr was able to generate low energy structured geometries for Janus clusters using a two-patched potential model for Janus particles where directional interaction was modulated to be strongest when similar patches directly point at each other and falls off as patches deviate further from perfect alignment. As far as the literature is concerned, this is the first time that IWD has been modified for atomic cluster optimization and combined with geometry operators to find the lowest energies for Janus clusters up to size 50 and on size 100.

Chapter 10

Conclusion and Recommendations

10.1 Conclusions of the Present Study

The nature-inspired global optimization algorithm, Modified Intelligent Water Drops with Perturbation Operators (MIWD+Perop and MIWD+CombiOp)[1] is developed and its application in finding the lowest energy molecular configuration have been studied in this doctoral thesis. Three archetypal test problems, Lennard-Jones, Binary Lennard-Jones and Morse clusters, were used to determine the performance of the algorithm. In addition, a potential model based on the LJ potential function with a modulated attraction term to describe the angular interaction of two-patched Janus particles is proposed. Using MIWD+CombiOp, for the first time on Janus clusters, the proposed Janus potential model is also optimized. The major contributions and conclusions of this study are summarized below:

1. An extensive review of global optimization algorithms with their applications in configurational optimization of Lennard-Jones (LJ), Binary Lennard-Jones (BLJ) and Morse Clusters is presented. Review of these algorithms were focused on initial configuration settings, relaxation algorithm used and potential energy surface traversal techniques. Numerical studies on Janus particles have also been reviewed with particular interest on potential models used to describe patchy particles.

2. The MIWD+PerOp and MIWD+CombiOp, which integrates a modified In-

[1]MIWD+PerOp is a two-step algorithm consisting of a modified Intelligent Water Drops Algorithm in the first phase and an iteration of a single perturbation operator in the second phase. MIWD+CombiOp, on the other hand, uses an iteration of two perturbation operators in the second phase.

telligent Water Drops (MIWD) algorithm (Phase 1) for suitability to atomic cluster optimization with various perturbation operators (Phase 2), are developed. Furthermore, when using the MIWD, there has been no need to provide *a priori* information about the geometry of the system to jump-start the algorithm as a result of the modifications to IWD. An efficient relaxation method, called L-BFGS, has been shown to provide significantly better outcomes as an aid to MIWD+PerOp/MIWD+CombiOp. Various perturbation operators to generate new clusters were both adapted and developed for this study. The effectiveness of the modified IWD together with various perturbation operators is assessed on known benchmark systems for atomic cluster optimization.

3. MIWD+PerOp results show that it is an effective unbiased optimization for finding the global optima of up to size $N = 104$ for a LJ system. MIWD+PerOp success rates on LJ clusters up to size $N = 104$, specially on well-known difficult cluster sizes, were comparatively higher than some published results.

4. MIWD+PerOp results on the more challenging Binary Lennard-Jones clusters up to size $N = 50$ for 6 values of atomic size ratio indicated an ability to find the global optima on most of the test instances. MIWD+CombiOp, which used a combination of perturbation operators in Phase 2 of the algorithm, was implemented and improved on the performance where MIWD+PerOp failed to find the GO. MIWD+PerOp and MIWD+CombiOp were able to find 293 out of the 300 instances tested under the Binary LJ system.

5. MIWD+PerOp results on Morse clusters, a tougher system for optimization due to presence of more varied structural behaviors in neighbouring cluster sizes, show robustness in rediscovering all GO up to size 60 under the long-range potential parameter $\alpha = 6.0$. MIWD+PerOp discovered most of the GO up to size 60 under the short-range potential parameter $\sigma = 14.0$ missing only 5 of the test instances.

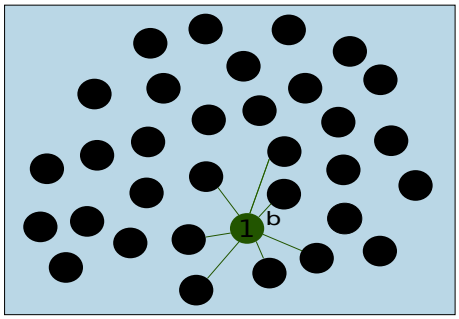
6. The Kern-Frenkel potential describing patchy hard sphere model of Janus particles has been widely used in numerical study of colloidal particles. Particles under the Kern-Frenkel potential, if patches are aligned within an allowable range, attract via a short-ranged non-smooth square-well potential. A different model for single-site particles used a modulated LJ potential which made the potential smooth/continuous as a function of the orientation of the particles. The former model, however, was insufficient to

describe the complex angular potential landscape which the latter provides but which was also inadequate to describe a two-patched/two-site particles where patch/site A (B) can only be attracted to patch/site A (B). In this study, the angular term used to modulate the LJ potential was transformed to suit two-patched/two-site Janus particles which allows : (1) patch A-A and patch B-B attractions but patch A-B repulsions; and (2) smooth functional decay or increase of angular term coefficient as the interparticle vector/axis deviates from perfect alignment. Using MIWD+CombiOp, the transformed model generated well-structured Janus clusters with identifiable building blocks across all tested cluster sizes namely: 3-particle triangle, 4-particle tetrahedron, skewed 5-particle trigonal bipyramid, 6-particle rectangular bipyramid, 7-particle pentagonal bipyramid, 8-particle irregular antiprism closed by two parallelograms and 9-particle irregular antiprism closed by a pentagon and a parallelgram. These building blocks were found to be embedded in the larger Janus particle clusters studied (up to size N=50).

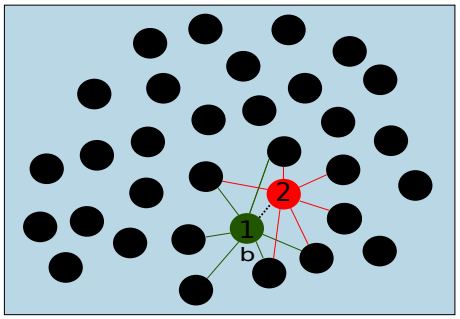
10.2 Recommendations for Future Work

Non-linear building of cluster for Phase 1 of MIWD+PerOp/MIWD+CombiOp : The current implementation of building a cluster for Phase 1 of MIWD+PerOp/MIWD+CombiOp is linear where the next particle to be added to the cluster must come from the recently added particle. This was the original implementation as this is more memory efficient in the long run. Another implementation could also be attempted where the building can emanate from any of the particles selected in a previous construction step. This idea would involve more calculations of probabilities and would need more memory allocation however the clusters that will be generated will more likely be more compact than the linear implementation. Figure 10.1 shows a schematic diagram of how the building of one cluster with 5 particles flows. The black circles represent particles scattered in 3D configurational space, the numbers on the circles represent a building step and the particle underneath the numbers added on that step, the lines between particles represent connectivity and a letter '**b**' beside the connection represents the connectivity with the best probability of selecting that connection.

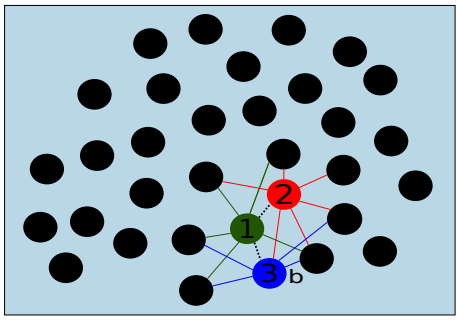
Parameter optimization for Phase 1 of MIWD+PerOp/MIWD+CombiOp : This study has widely used arbitrary parameter settings and some settings from previous implementations of Intelligent Water Drops and Lennard-Jones cluster optimization which were



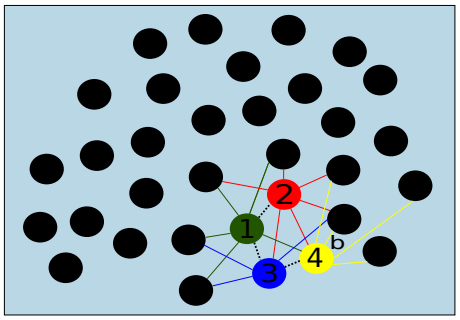
(a) Step 1



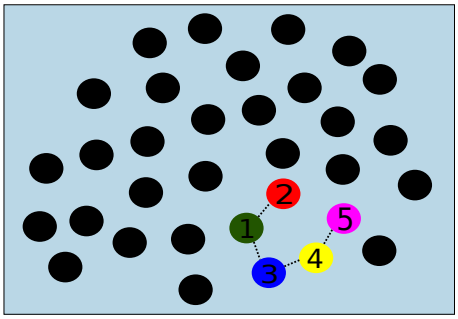
(b) Step 2



(c) Step 3



(d) Step 4



(e) Step 5

Figure 10.1: Proposed non-linear building of clusters for Phase 1 of MIWD+PerOp/MIWD+CombiOp where growth of the solution/cluster can come from any particle already chosen by the IWD. At each step, probabilities are calculated from all particles in the IWD to all other particle sites not yet visited. The particle connection associated with the best, $\boldsymbol{b}$, of all these probabilities is eventually included into the cluster. For visual clarity, the connections are only drawn for IWD particles to neighbouring, unvisited particles but probability calculations should be done between all pairs. Dotted lines show the connection of included particles into the IWD.

optimized for the experiments in their studies. A sensitivity analysis of parameter settings would lead to improvement in results for Phase 1.

Appendix A

Limited Memory BFGS

The Limited-Memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) as described by Liu and Nocedal [Liu & Nocedal, 1989] is a method that solves the unconstrained minimization problem,

$$MinF(x), x = (x_1, x_2, \ldots, x_n) \tag{A.1}$$

if the objective function $F(x)$ and its gradient $G(x)$ are computable. L-BFGS is a quasi-Newton implementation that replaces computation of the inverse of the Hessian matrix of the objective function with an approximation. The computational cost for calculating the inverse Hessian matrix is high for minimization problems involving a large number of variables. L-BFGS thus uses an approximation to the inverse Hessian derived from past m updates of x and the gradient $\nabla f(x)$. The algorithm below gives a precise description of the L-BFGS as presented in Liu and Nocedal [Liu & Nocedal, 1989].

`L-BFGS Method`

1. Choose x_0, m, $0 < \beta' < \frac{1}{2}$, $\beta' < \beta < 1$, and a symmetric and positive definite starting matrix H_0. Set $k = 0$.

2. Compute

$$d_k = -H_k g_k \tag{A.2}$$

$$x_{k+1} = x_k + \alpha_k d_k \tag{A.3}$$

where α_k satisfies the *Wolfe* conditions:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \beta' \alpha_k g_k^T d_k \tag{A.4}$$

$$g(x_k + \alpha_k d_k)^T d_k \geq \beta g_k^T d_k \tag{A.5}$$

Steplength $\alpha_k = 1$ first.

3. Let $\hat{m} = min\{k, m-1\}$. Update H_0 $\hat{m}+1$ times using the pairs $\{y_j, s_j\}_{j=k-\hat{m}}^{k}$, i.e. let

$$\begin{aligned} H_{k+1} = & \left(V_k^T \dots V_{k-\hat{m}}^T\right) H_0 \left(V_{k-\hat{m}} \dots V_k\right) \\ & + \rho_{k-\hat{m}} \left(V_k^T \dots V_{k-\hat{m}+1}^T\right) s_{k-\hat{m}} s_{k-\hat{m}}^T \left(V_{k-\hat{m}+1} \dots V_k\right) \\ & + \rho_{k-\hat{m}+1} \left(V_k^T \dots V_{k-\hat{m}+2}^T\right) s_{k-\hat{m}+1} s_{k-\hat{m}+1}^T \left(V_{k-\hat{m}+2} \dots V_k\right) \\ & \vdots \\ & + \rho_k s_k s_k^T. \end{aligned} \tag{A.6}$$

4. Set $k := k + 1$ and go to step 2.

The iterates in the above algorithm are denoted by x_k and $x_{k+1} - x_k$ and $g_{k+1} - g_k$ are defined by s_k and y_k, respectively. The inverse Hessian is approximated by the following formula:

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T, \tag{A.7}$$

where $p_k = \frac{1}{y_k^T s_k}$, and $V_k = I - \rho_k s_k s_k^T$.

Both the original Fortran code [LBF, n.d.] and C port [Okazaki, 2012] of the L-BFGS implementation were used in this study.

Appendix B

Random_Seed

In a Fortran program, calling the RANDOM_NUMBER subroutine several times generates a sequence of numbers which are distributed randomly. However, it is likely that this sequence repeats upon running the program again. Generating random numbers has been a big component of the processes involved in this study and thus a reliable *true* random number generator is desirable. A module published online [Stevenson, 2012] by the University of Surrey provides a portable way to get different random sequence of numbers every time a program is run. This was done by changing the seed of every random sequence of numbers generated using the RANDOM_SEED intrinsic subroutine. The code below shows the program as presented in the University of Surrey module webpage. A detailed description follows.

```
PROGRAM ranseed
  IMPLICIT NONE
  ! ----- variables for portable seed setting -----
  INTEGER :: i_seed
  INTEGER, DIMENSION(:), ALLOCATABLE :: a_seed
  INTEGER, DIMENSION(1:8) :: dt_seed
  ! ----- end of variables for seed setting -----
  REAL :: r
  ! ----- Set up random seed portably -----
  CALL RANDOM_SEED(size=i_seed)
  ALLOCATE(a_seed(1:i_seed))
  CALL RANDOM_SEED(get=a_seed)
  CALL DATE_AND_TIME(values=dt_seed)
  a_seed(i_seed)=dt_seed(8); a_seed(1)=dt_seed(8)*dt_seed(7)*dt_seed(6)
  CALL RANDOM_SEED(put=a_seed)
  DEALLOCATE(a_seed)
```

```
  ! ----- Done setting up random seed -----
  CALL RANDOM_NUMBER(r)
  WRITE(6,*) 'random number is ',r
END PROGRAM ranseed
```

The seed value is "seeded" by the intrinsic subroutine DATE_AND_TIME which gives date and time information up to milliseconds of a second thereby greater chances of acquiring different seeds everytime the subroutine is called. The Fortran standard implements a random seed which is a 1-dimensional array of integers however the size of the array is determined by the compiler. Fortunately, there is a way to determine this through the *size* argument of the subroutine RANDOM_SEED. This value is then used to allocate another integer array of the same size and assign the values of the seed using the *get* argument of RANDOM_SEED.

A call to the subroutine DATE_AND_TIME subroutine is then made gathering the following information : 4-digit year, month of the year, day of the month, time difference with respect to UTC in minutes, hour of the day, minutes of the hour, second of the minute and milliseconds of the second. The milliseconds value is then set to the final element of the seed array while the product of the minutes, seconds and milliseconds is set to the first element. These two values are expected to different all times the routine is called. The updated array is written back (using *set* argument of RANDOM_SEED). Subsequent calls to the RANDOM_NUMBER subroutine is guaranteed to get a sequence of pseudorandom numbers which is different every time the program is run.

The random number generator above has been guaranteed to work on every computer-compiler combination and has been successfully applied in this study to generate random numbers for initializing particle positions and for random processes involved in the perturbation operators of the algorithm.

Appendix C

NAG OPT_CONJ_GRAD (e04dgc)

The Numerical Algorithms Group (NAG) Opt_Conj_Grad [NAG, n.d.] includes a pre-conditioned, limited-memory quasi-Newton conjugate gradient method for handling minimization of nonlinear functions of several variables. Test runs in this study, specifically under the Lennard-Jones system, initially used a function from the NAG C Library called *e04dgc*.
The algorithm proceeds as follows:

It starts with a given starting point x_0 with k (iteration number) at 0. Each iteration requires the gradient vector g_k evaluated at x_k, the *kth* estimate of the minimum. At each iteration a vector p_k (the direction of search) is computed and the new estimate $x_{k+1} = x_k + \alpha_k p_k$ is generated where the step length α_k minimizes the function $F(x_k + \alpha_k p_k)$ with respect to the scalar α_k. At the start of each line search, an initial approximation α_0 to the step α_k is taken as :

$$\alpha_0 = min\{1, 2|F_k - F_{est}|/g_k^T g_k\}$$

where F_{est} is a user-supplied estimate of the function value at the solution. If not specified, α_0 is set to 1.0. Subsequent step length estimates are computed using cubic interpolation with safeguards.

A quasi-Newton methods then computes the search direction p_k by updating the inverse of the approximate Hessian H_k and computing

$$p_{k+1} = -H_{k+1} g_{k+1} \tag{C.1}$$

The updating formula for the approximate inverse is given by Equation C.2 where $y_k = g_{k-1} - g_k$ and $s_k = x_{k+1} - x_k = \alpha_k p_k$.

$$H_{k+1} = H_k - \frac{1}{y_k^T s_k}\left(H_k y_k s_k^T + y_k^T s_k H_k\right) + \frac{1}{y_k^T s_k}\left(1 + \frac{y_k^T H_k y_k}{y_k^T s_k}\right) s_k s_k^T \qquad \text{(C.2)}$$

NAG Opt_Conj_Grad uses a two-step method with restarts and preconditioning. Using a limited-memory quasi-Newton formula guarantees p_{k+1} to be a descent direction if all the inner products $y_k^T s_k$ are positive for all the vectors y_k and s_k used in the updating formula.

The termination criteria of NAG Opt_Conj_Grad are as follows: Let τ_F be a parameter indicating the correct number of figures desired in F_k. If the following three conditions are satisfied, then the algorithm is considered to have converged.

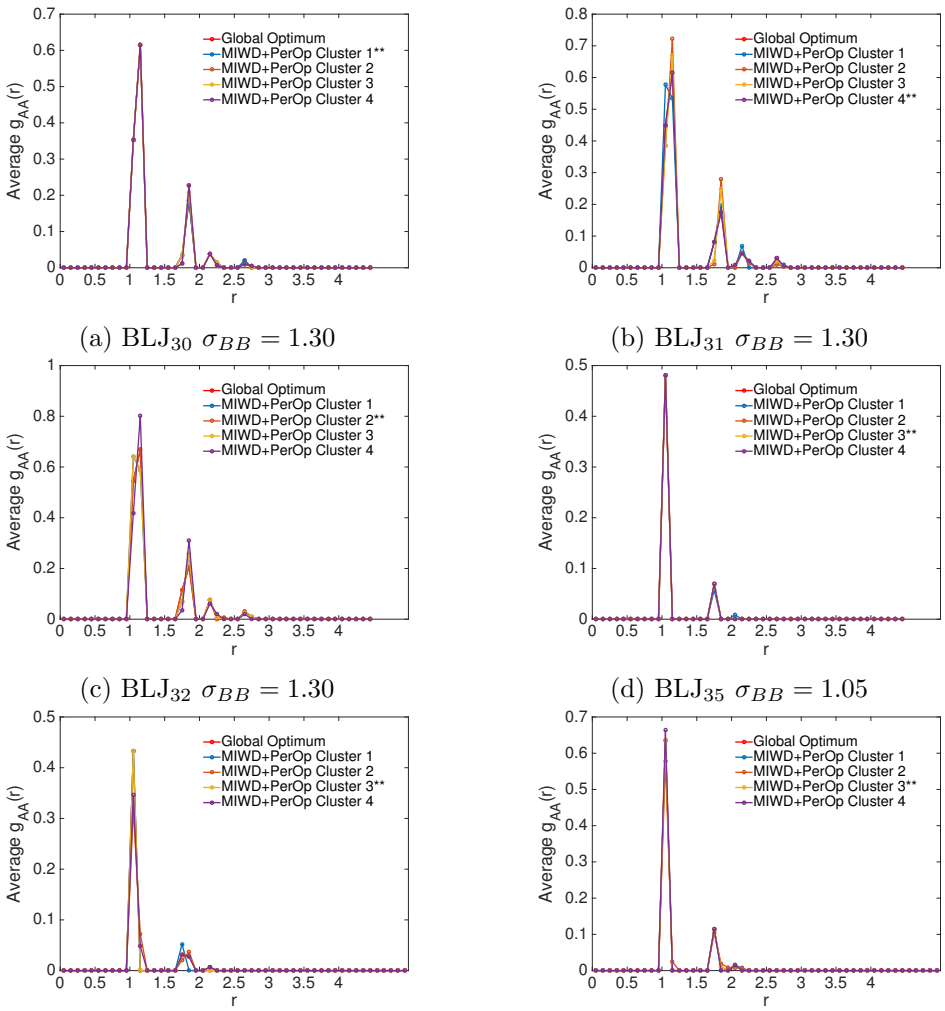
i $F_{k-1} - F_k < \tau_F\left(1 + |F_k|\right)$

ii $\|x_{k-1} - x_k\| < \sqrt{\tau_F}\left(1 + \|x_k\|\right)$

iii $\|g_k\| \leq \tau_F^{1/3}\left(1 + |F_k|\right)$ or $\|g_k\| < \varepsilon_A$, where ε_A is the absolute error associated with computing the objective function.

Appendix D

Average RDF of MIWD+PerOp results for selected Binary Lennard Jones Clusters



(a) BLJ_{30} $\sigma_{BB} = 1.30$

(b) BLJ_{31} $\sigma_{BB} = 1.30$

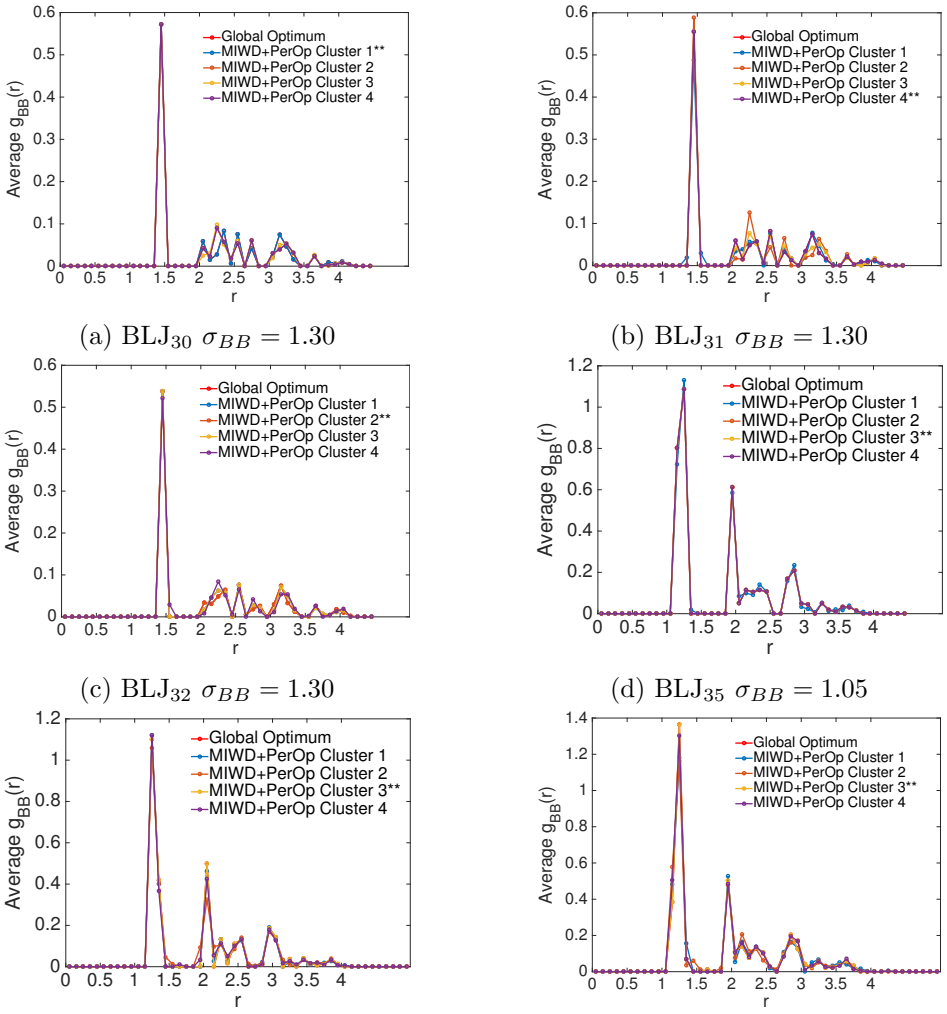
(c) BLJ_{32} $\sigma_{BB} = 1.30$

(d) BLJ_{35} $\sigma_{BB} = 1.05$

(e) BLJ_{35} $\sigma_{BB} = 1.15$

(f) BLJ_{40} $\sigma_{BB} = 1.05$

Figure D.1: Average RDF of MIWD+Knead results compared to CCD putative global optima for selected test instances. Clusters marked with ** achieved the putative GO. The plots above show the average probability of finding type A particle on a particular shell from a given reference type A particle.



(a) BLJ_{30} $\sigma_{BB} = 1.30$

(b) BLJ_{31} $\sigma_{BB} = 1.30$

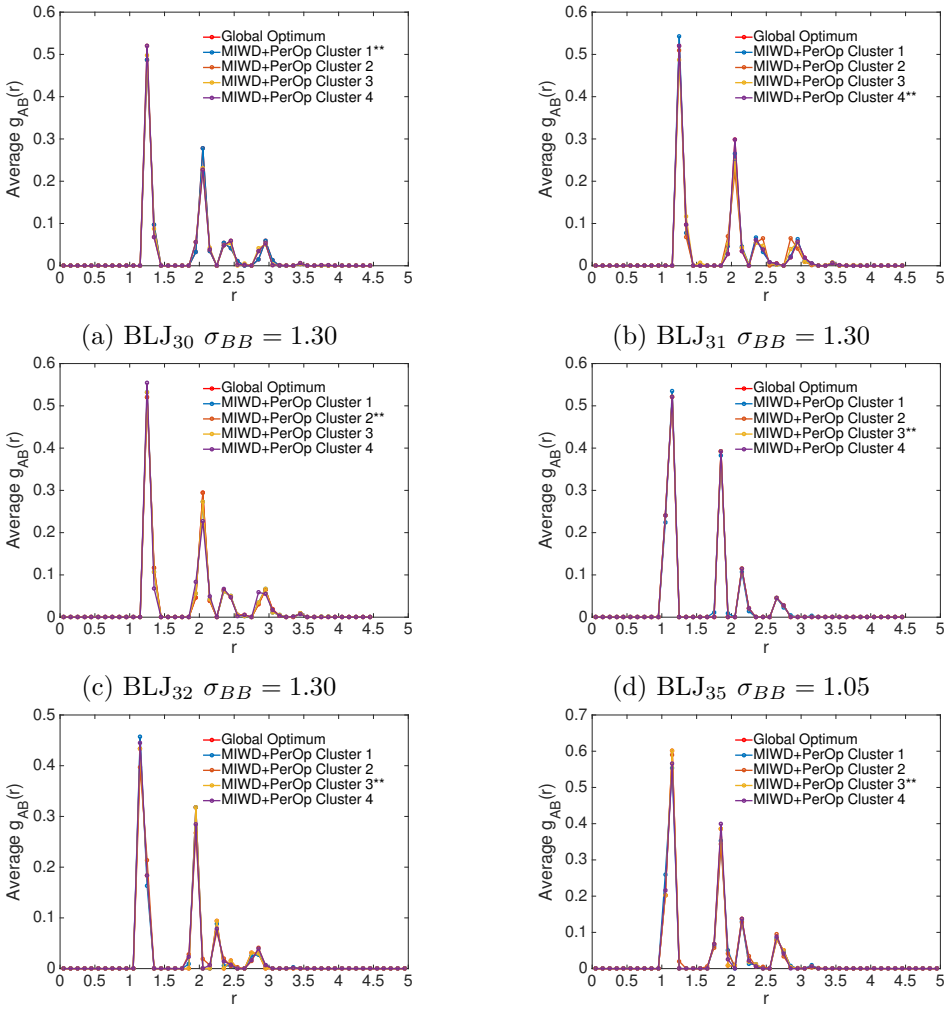
(c) BLJ_{32} $\sigma_{BB} = 1.30$

(d) BLJ_{35} $\sigma_{BB} = 1.05$

(e) BLJ_{35} $\sigma_{BB} = 1.15$

(f) BLJ_{40} $\sigma_{BB} = 1.05$

Figure D.2: Average RDF of MIWD+Knead results compared to CCD putative global optima for selected test instances. Clusters marked with ** achieved the putative GO. The plots above show the average probability of finding type B particle on a particular shell from a given reference type B particle.



(a) BLJ_{30} $\sigma_{BB} = 1.30$

(b) BLJ_{31} $\sigma_{BB} = 1.30$

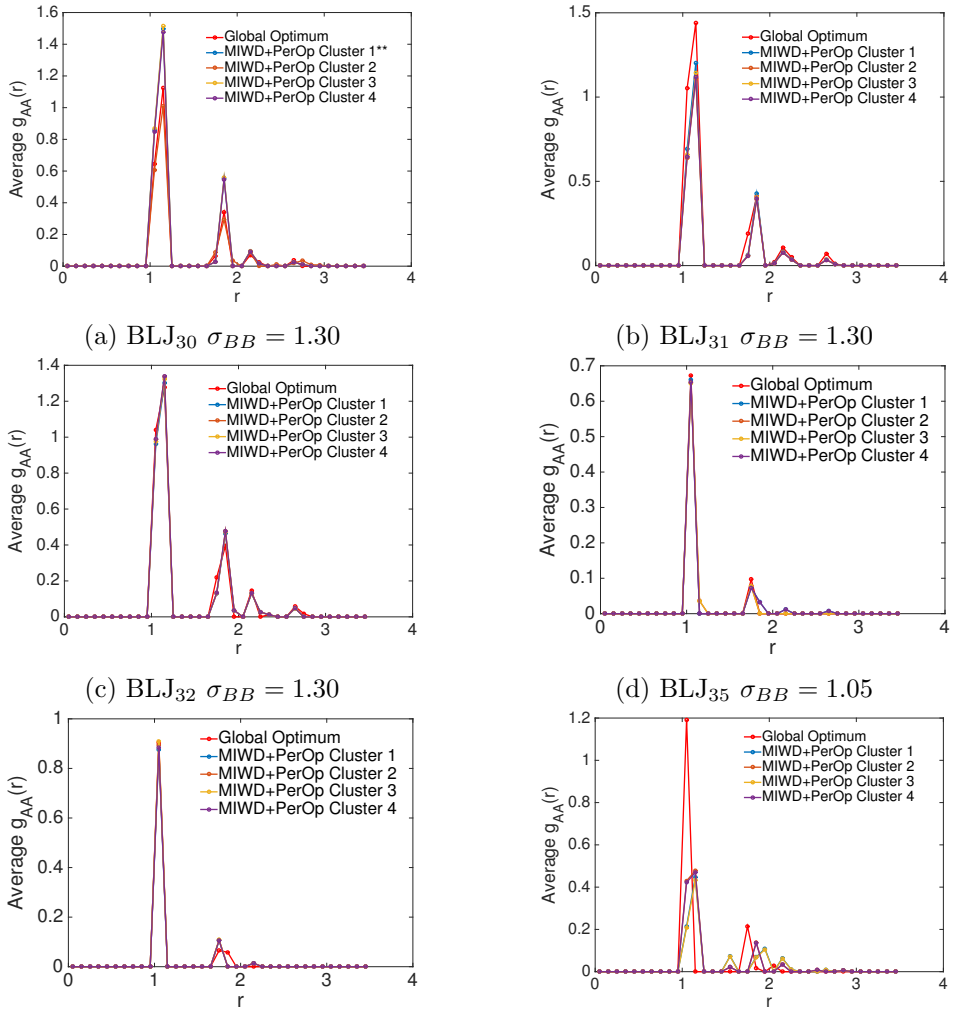
(c) BLJ_{32} $\sigma_{BB} = 1.30$

(d) BLJ_{35} $\sigma_{BB} = 1.05$

(e) BLJ_{35} $\sigma_{BB} = 1.15$

(f) BLJ_{40} $\sigma_{BB} = 1.05$

Figure D.3: Average RDF of MIWD+Knead results compared to CCD putative global optima for selected test instances. Clusters marked with ** achieved the putative GO. The plots above show the average probability of finding type B particle on a particular shell from a given reference type A particle.



(a) BLJ_{30} $\sigma_{BB} = 1.30$

(b) BLJ_{31} $\sigma_{BB} = 1.30$

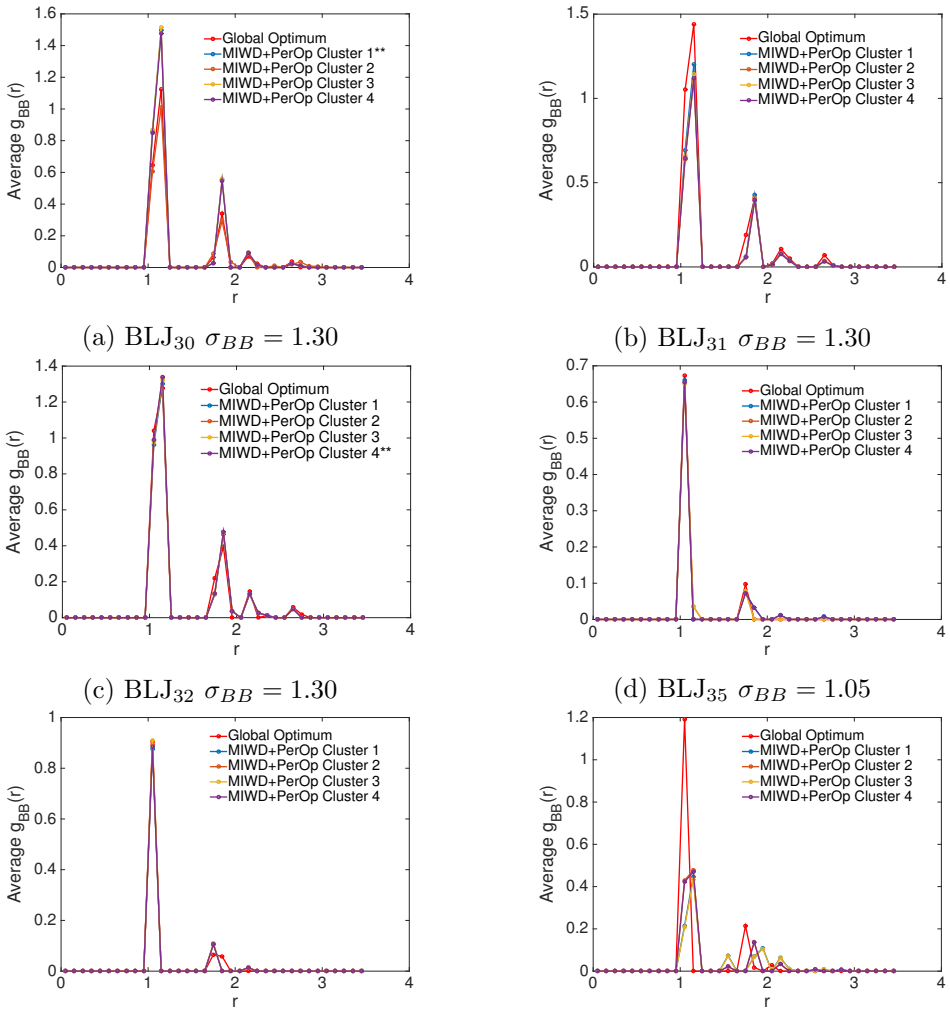
(c) BLJ_{32} $\sigma_{BB} = 1.30$

(d) BLJ_{35} $\sigma_{BB} = 1.05$

(e) BLJ_{35} $\sigma_{BB} = 1.15$

(f) BLJ_{40} $\sigma_{BB} = 1.05$

Figure D.4: Average RDF of MIWD+CutSpliceVar suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type A particle on a particular shell from a given reference type A particle.



(a) BLJ_{30} $\sigma_{BB} = 1.30$

(b) BLJ_{31} $\sigma_{BB} = 1.30$

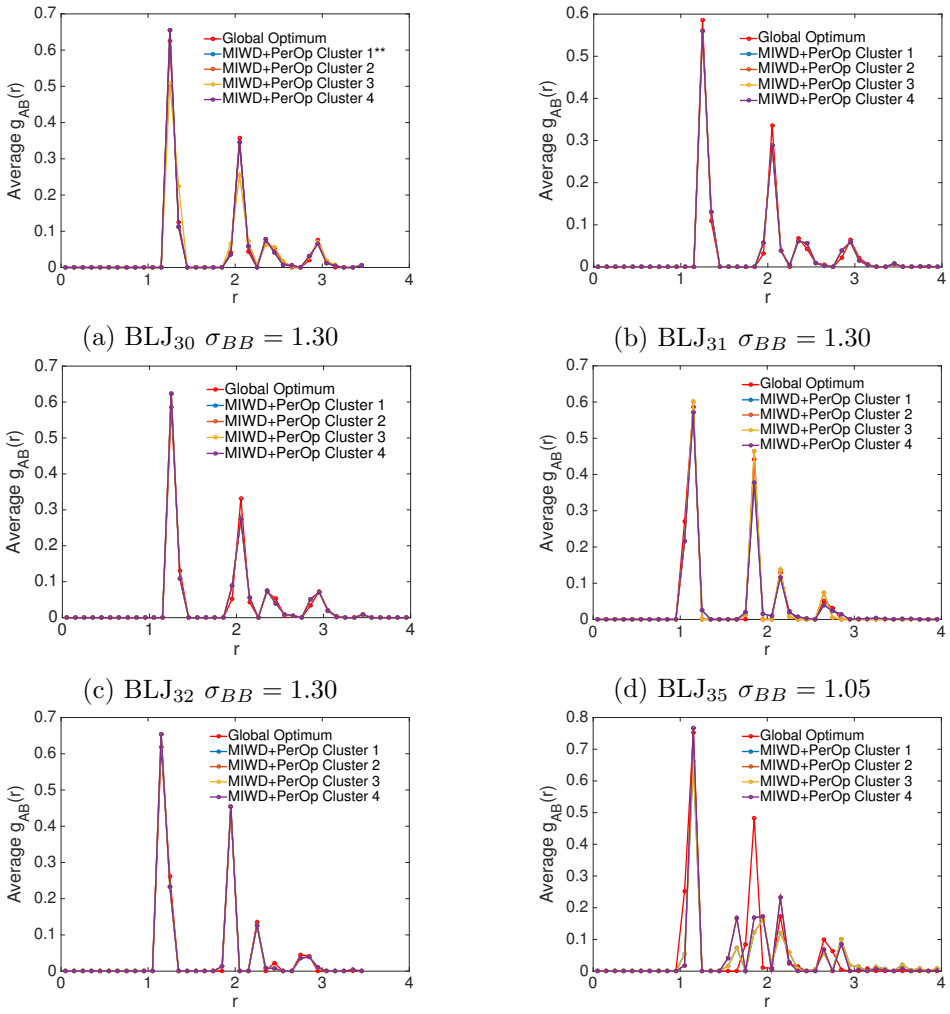
(c) BLJ_{32} $\sigma_{BB} = 1.30$

(d) BLJ_{35} $\sigma_{BB} = 1.05$

(e) BLJ_{35} $\sigma_{BB} = 1.15$

(f) BLJ_{40} $\sigma_{BB} = 1.05$

Figure D.5: Average RDF of MIWD+CutSpliceVar suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type B particle on a particular shell from a given reference type B particle.



(a) BLJ_{30} $\sigma_{BB} = 1.30$

(b) BLJ_{31} $\sigma_{BB} = 1.30$

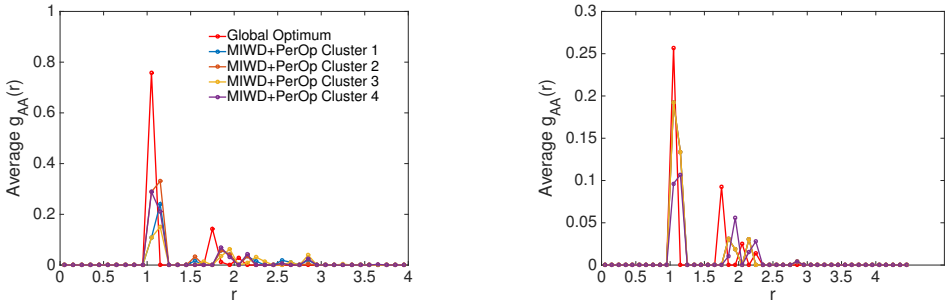
(c) BLJ_{32} $\sigma_{BB} = 1.30$

(d) BLJ_{35} $\sigma_{BB} = 1.05$

(e) BLJ_{35} $\sigma_{BB} = 1.15$

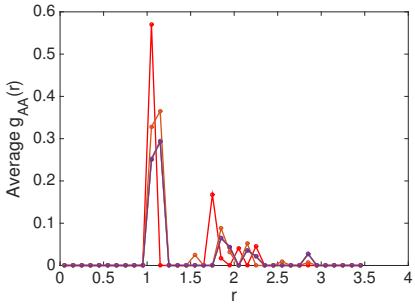
(f) BLJ_{40} $\sigma_{BB} = 1.05$

Figure D.6: Average RDF of MIWD+CutSpliceVar suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type B particle on a particular shell from a given reference type A particle.

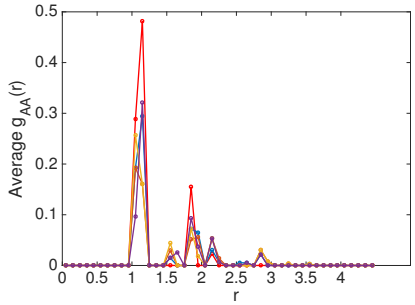


(a) BLJ_{41} $\sigma_{BB} = 1.05$

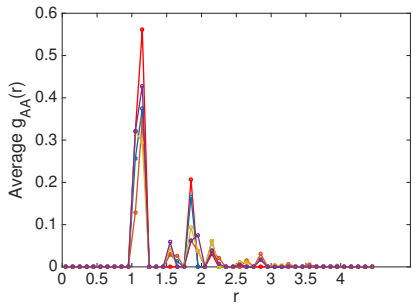
(b) BLJ_{43} $\sigma_{BB} = 1.05$



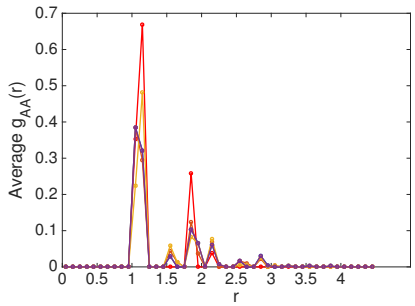
(c) BLJ_{45} $\sigma_{BB} = 1.05$



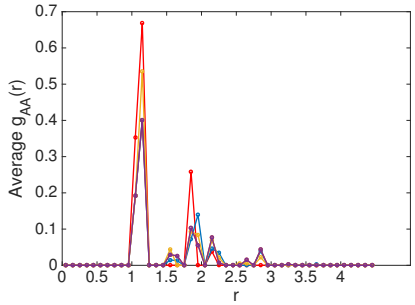
(d) BLJ_{46} $\sigma_{BB} = 1.05$



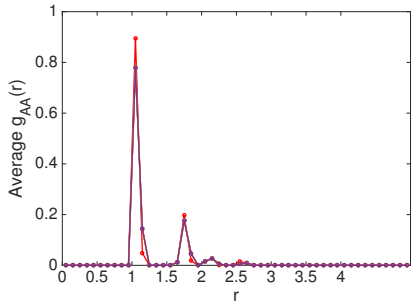
(e) BLJ_{47} $\sigma_{BB} = 1.05$



(f) BLJ_{48} $\sigma_{BB} = 1.05$

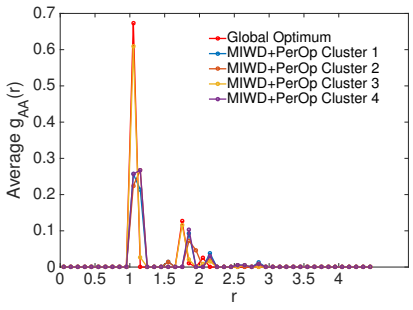


(g) BLJ_{49} $\sigma_{BB} = 1.05$

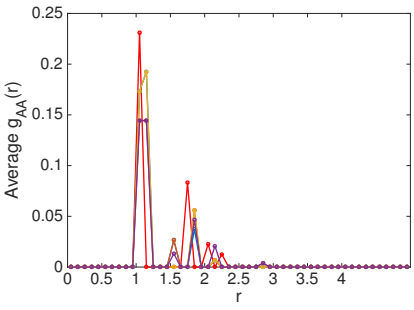


(h) BLJ_{47} $\sigma_{BB} = 1.10$

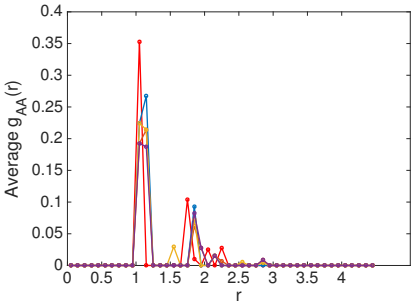
Figure D.7: Average RDF of MIWD+CutSpliceVar suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type A particle on a particular shell from a given reference type A particle.



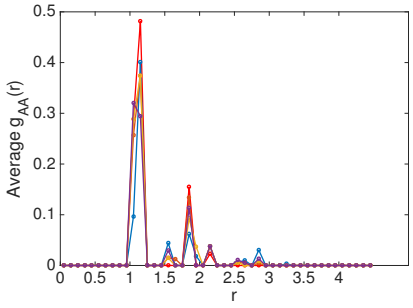
(a) BLJ_{41} $\sigma_{BB} = 1.05$



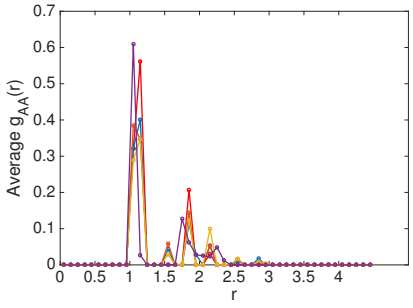
(b) BLJ_{43} $\sigma_{BB} = 1.05$



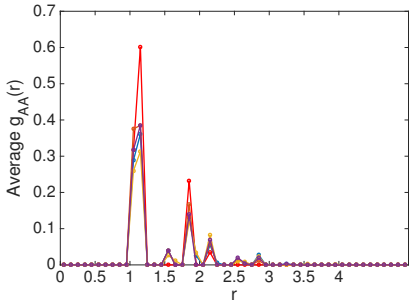
(c) BLJ_{45} $\sigma_{BB} = 1.05$



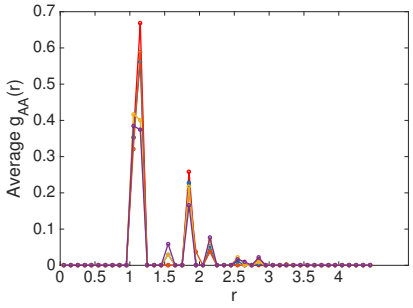
(d) BLJ_{46} $\sigma_{BB} = 1.05$



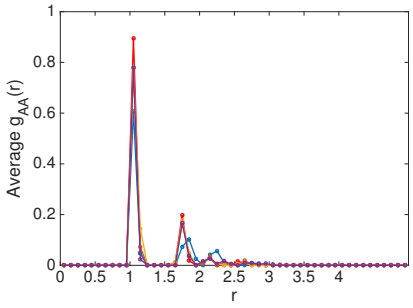
(e) BLJ_{47} $\sigma_{BB} = 1.05$



(f) BLJ_{48} $\sigma_{BB} = 1.05$

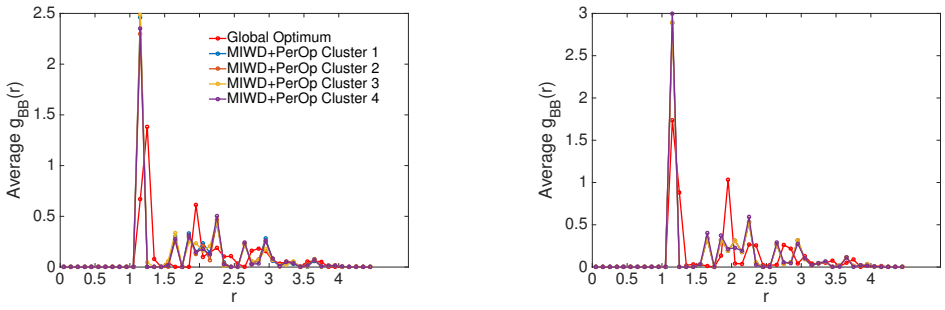


(g) BLJ_{49} $\sigma_{BB} = 1.05$



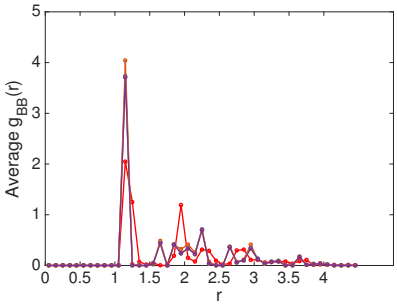
(h) BLJ_{47} $\sigma_{BB} = 1.10$

Figure D.8: Average RDF of MIWD+Knead suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type A particle on a particular shell from a given reference type A particle.

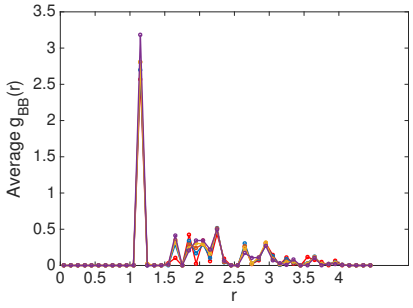


(a) BLJ_{41} $\sigma_{BB} = 1.05$

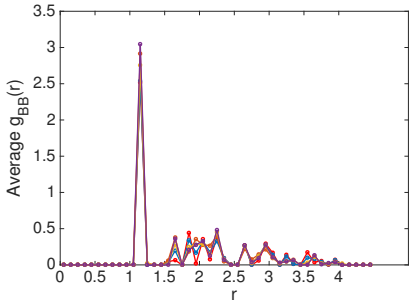
(b) BLJ_{43} $\sigma_{BB} = 1.05$



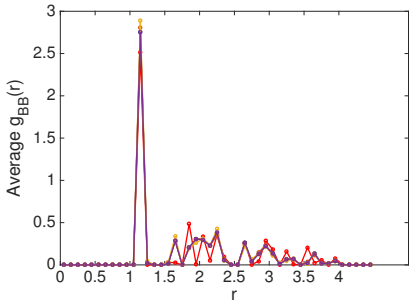
(c) BLJ_{45} $\sigma_{BB} = 1.05$



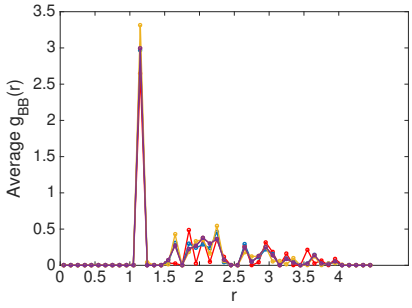
(d) BLJ_{46} $\sigma_{BB} = 1.05$



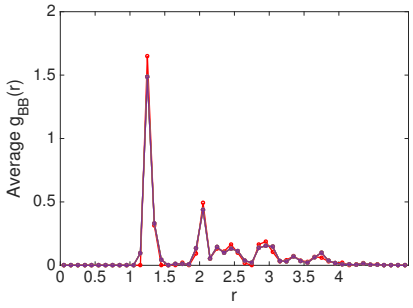
(e) BLJ_{47} $\sigma_{BB} = 1.05$



(f) BLJ_{48} $\sigma_{BB} = 1.05$

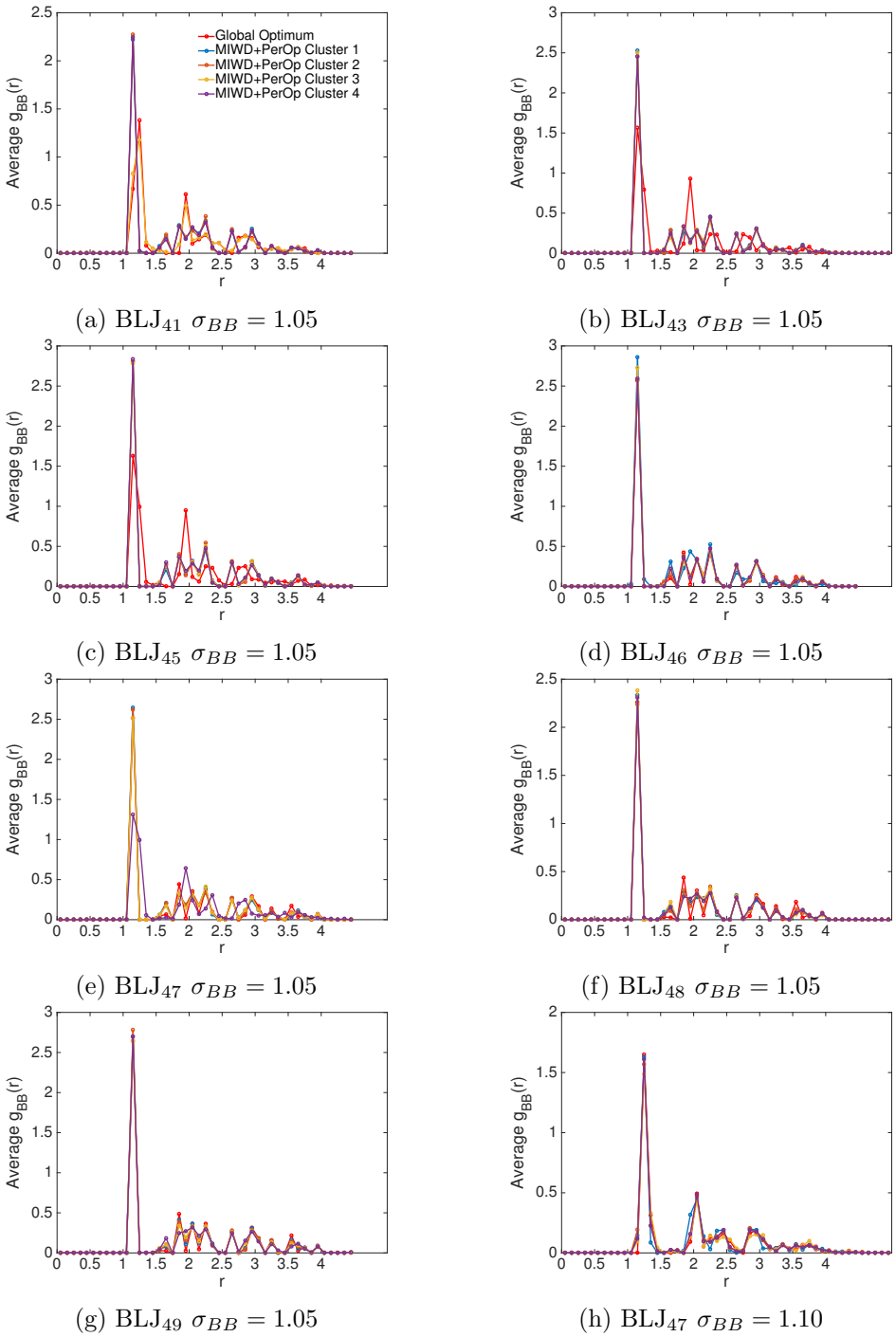


(g) BLJ_{49} $\sigma_{BB} = 1.05$



(h) BLJ_{47} $\sigma_{BB} = 1.10$

Figure D.9: Average RDF of MIWD+CutSpliceVar suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type B particle on a particular shell from a given reference type B particle.



(a) BLJ_{41} $\sigma_{BB} = 1.05$

(b) BLJ_{43} $\sigma_{BB} = 1.05$

(c) BLJ_{45} $\sigma_{BB} = 1.05$

(d) BLJ_{46} $\sigma_{BB} = 1.05$

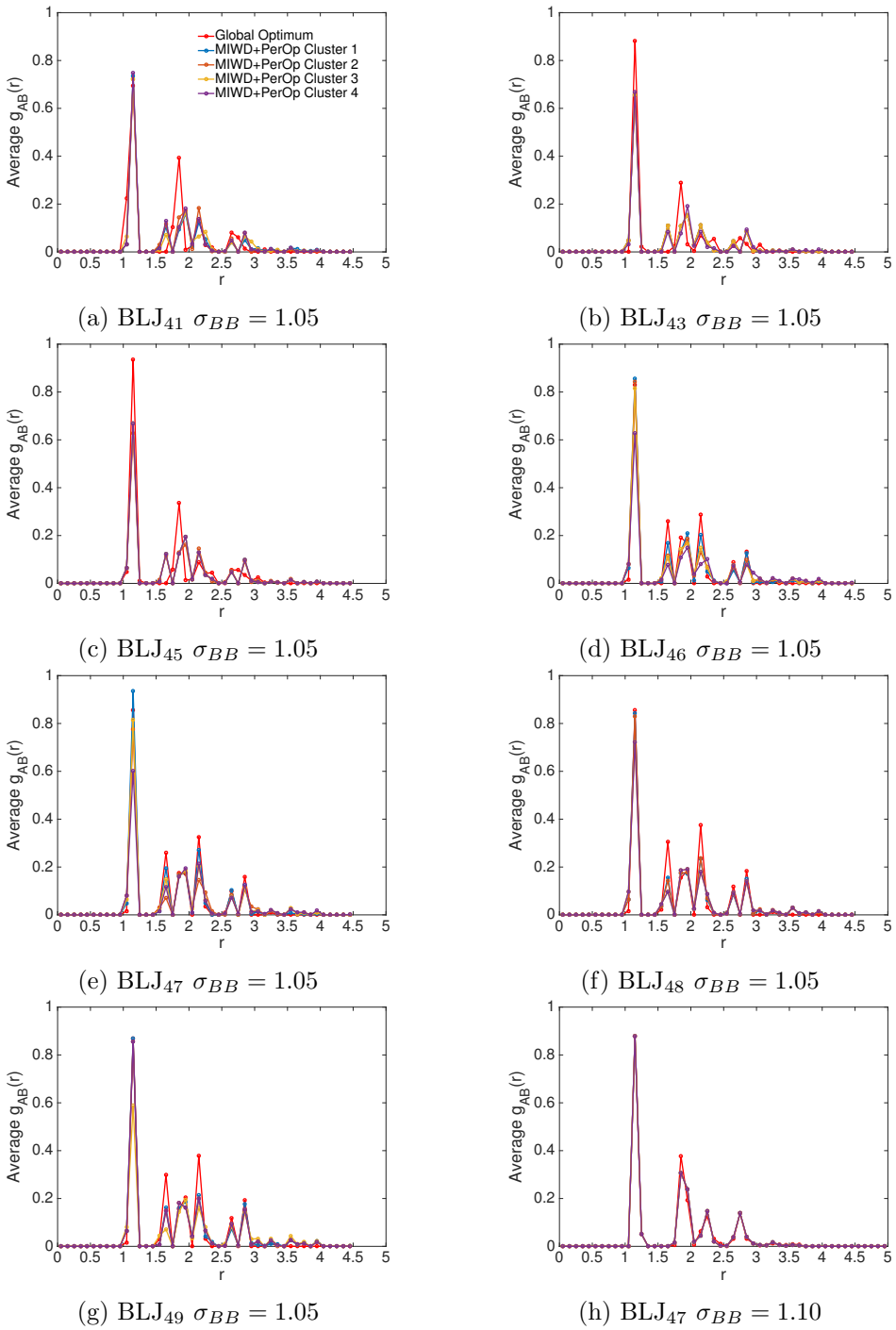
(e) BLJ_{47} $\sigma_{BB} = 1.05$

(f) BLJ_{48} $\sigma_{BB} = 1.05$

(g) BLJ_{49} $\sigma_{BB} = 1.05$

(h) BLJ_{47} $\sigma_{BB} = 1.10$

Figure D.10: Average RDF of MIWD+Knead suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type B particle on a particular shell from a given reference type B particle.



(a) BLJ_{41} $\sigma_{BB} = 1.05$

(b) BLJ_{43} $\sigma_{BB} = 1.05$

(c) BLJ_{45} $\sigma_{BB} = 1.05$

(d) BLJ_{46} $\sigma_{BB} = 1.05$

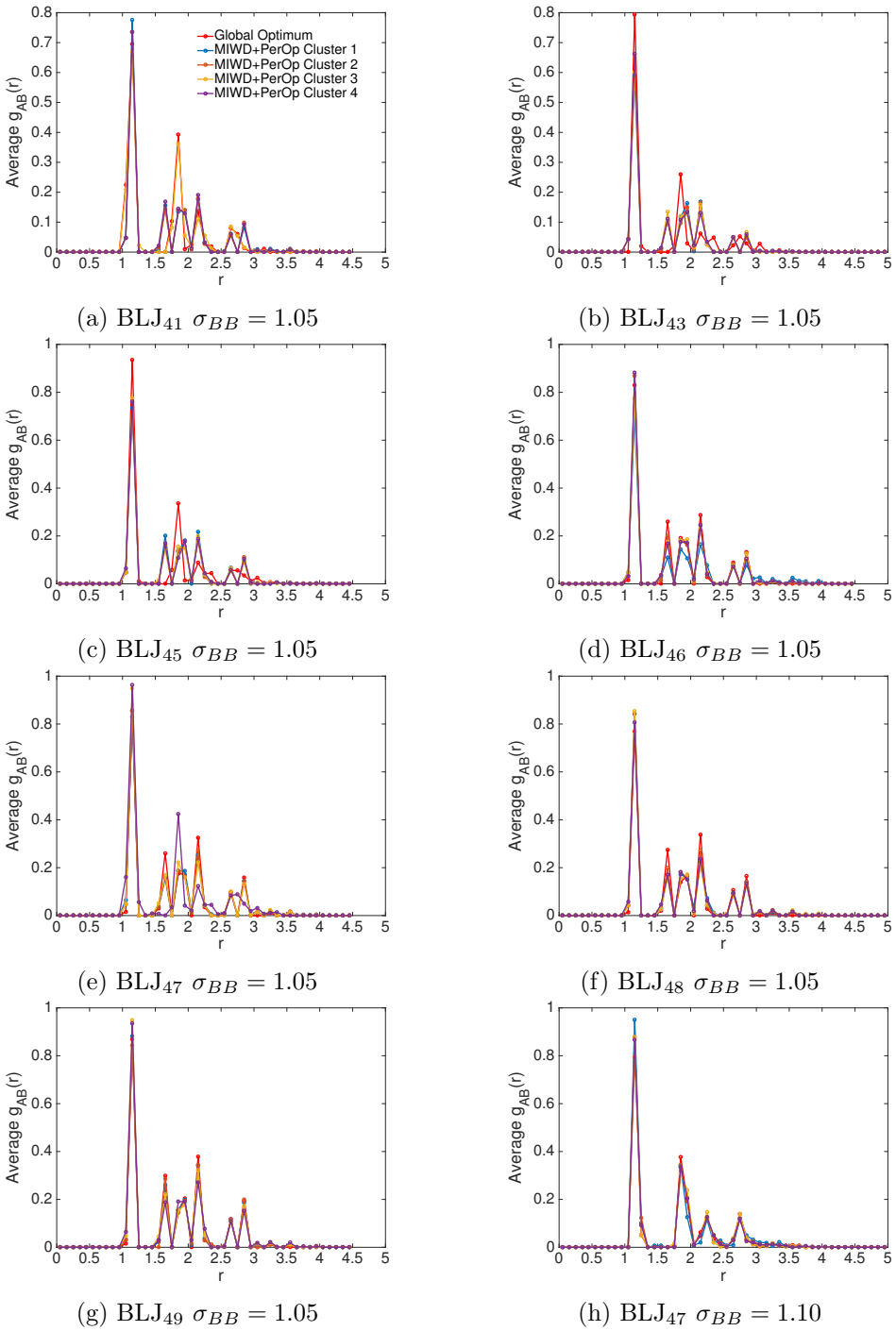
(e) BLJ_{47} $\sigma_{BB} = 1.05$

(f) BLJ_{48} $\sigma_{BB} = 1.05$

(g) BLJ_{49} $\sigma_{BB} = 1.05$

(h) BLJ_{47} $\sigma_{BB} = 1.10$

Figure D.11: Average RDF of MIWD+CutSpliceVar suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type A particle on a particular shell from a given reference type B particle.



(a) BLJ_{41} $\sigma_{BB} = 1.05$

(b) BLJ_{43} $\sigma_{BB} = 1.05$

(c) BLJ_{45} $\sigma_{BB} = 1.05$

(d) BLJ_{46} $\sigma_{BB} = 1.05$

(e) BLJ_{47} $\sigma_{BB} = 1.05$

(f) BLJ_{48} $\sigma_{BB} = 1.05$

(g) BLJ_{49} $\sigma_{BB} = 1.05$

(h) BLJ_{47} $\sigma_{BB} = 1.10$

Figure D.12: Average RDF of MIWD+Knead suboptimal results compared to CCD putative global optima for selected test instances. The plots above show the average probability of finding type B particle on a particular shell from a given reference type A particle.

Bibliography

The NAG C Library. The Numerical Algorithms Group (NAG), Oxford, United Kingdom.

Ahmad, T., Jameel, A., & Ahmad, B. 2011. Pattern Recognition using Statistical and Neural Techniques. *Computer Networks and Information Technology*, 87–91.

Alijla, B., Wong, L. P., Lim, C. P., Khader, A. T., & Al-Betar, M. A. 2014. A modified Intelligent Water Drops algorithm and its application to optimization problems. *Expert Systems with Applications*, **41**, 6555–6569.

Andretta, M., & Birgin, E. G. 2013. Deterministic and Stochastic Global Optimization Techniques for planar covering with ellipses problems. *European Journal of Operational Research*, **224**, 23–40.

Bah, T., Gould, T., Cruz, J., Harrington, B., Andler, J., Kosinski, K., & Owens, M. *Inkscape.*

Barron, C., Gomez, S., & Romero, D. 1997. Lower Energy Icosahedral Atomic Clusters with Incomplete Core. *Applied Math Letters*, **10**, 25–28.

Benthien, G. *Fortran Character String Utilities.* Available at `gbenthien.net/strings/`.

Blake, A. 1989. Comparison of the Efficiency of Deterministic and Stochastic Algorithms for Visual Reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **2**, 2–12.

Blake, A., & Zisserman, A. 1987. *Visual Reconstruction.* Cambridge, MA, USA: MIT Press.

Blum, C., & Roli, A. 2003. Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, **35**, 268–308.

Bytheway, I., & Kepert, D. L. 1992. The Mathematical Modelling of Cluster Geometry. *Journal of Mathematical Chemistry*, **9**, 161–180.

Cai, W., & Shao, X. 2002. A Fast Annealing Evolutionary Algorithm for Global Optimization. *Journal of Computational Chemistry*, **23**(4), 427–435.

Cai, W., Jiang, H., & Shao, X. 2002a. Global Optimization of Lennard-Jones Clusters by a Parallel Fast Annealing Evolutionary Algorithm. *Journal of Chemical Information and Computer Sciences*, **42**, 1099–1103.

Cai, W., Feng, Y., Shao, X., & Pan, Z. 2002b. Optimization of Lennard-Jones atomic clusters. *Journal of Molecular Structure (Theochem)*, **579**(1-3), 229–234.

Casagrande, C., & Veyssie, M. 1988. Janus beads-realization and 1st observation of interfacial properties. *Comptes Rendus de l'Academie des Sciences*, **306**, 1423–1425.

Cassioli, A., Locatelli, M., & Schoen, F. 2009. Global Optimization of binary Lennard-Jones clusters. *Optimization Methods and Software*, **24**(4-5), 819–835.

Chen, Y., Cui, Z., & Zeng, J. 2010. Structural Optimization of Lennard-Jones Clusters by Hybrid Social Cognitive Optimization Algorithm. *Pages 204–208 of: Proceedings of the 9th IEEE Conf on Cognitive Informatics.*

Colorni, A., Dorigo, M., & Maniezzo, V. 1991. Ditributed Optimization by Ant Colonies. *Pages 134–142 of: Proceedings of the European Conference on Artificial Life*, vol. 142.

Cui, Z., & Cai, X. 2010. Using Social Cognitive Optimization Algorithm to Solve Nonlinear Equations. *Proceedings of the 9th IEEE International Conference on Cognitive Informatics*, 199–203.

Deaven, D.M., & Ho, K.M. 1995. Molecular Geometry Optimization with a Genetic Algorithm. *Physical Review Letters*, **75**(2), 288–291.

Deaven, D.M., Morris, J.M., Ho, K.M., & Tit, N. 1996. Structural Optimization of Lennard-Jones clusters by genetic algorithm. *Chemical Physics Letters*, **256**, 195–200.

Deep, K., Katiyar, V., & Shashi. 2011. Minimizing Lennard-Jones potential using a real coded Genetic Algorithm and Particle Swarm Optimization. *World Journal of Modelling and Simuation*, **7**, 312–320.

Doerner, K. F., Hartl, R. F., & Reimann, M. 2003. CompetAnts for problem solving. *Central European Journal of Operations Research*, **11**, 115–141.

Dorigo, M., Maniezzo, V., & Colorni, A. 1996. Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, **26**, 29–41.

Doye, J., & Meyer, L. 2005. Mapping the Magic Numbers in Binary Lennard-Jones Clusters. *Physical Review Letters*, **95**(6), 063401.

Doye, J. P. K., & Meyer, L. 2006. The structure of binary Lennard-Jones clusters: The effects of atomic size ratio. *arXiv:cond-mat/0604250*.

Doye, J. P. K., & Wales, D. J. 1996. The effect of the range of the potential on the structures and stability of simple liquids: From clusters to buulk, from Sodium to C60. *Journal of Physics*, **29**, 4859–4894.

Doye, J. P. K., Wales, D. J., & Berry, R. S. 1995. The effect of the range of the potential on the structures of clusters. *The Journal of Chemical Physics*, **103**(10), 4234–4249.

Doye, J. P. K., Leary, R. H., Locatelli, M., & Schoen, F. 2004. The global optimization of Morse clusters by potential energy transformations. *INFORMS Journal on Computing*, **16**, 371–379.

Doye, J. P. K., Louis, A., Lin, I., Allen, L., Noya, E., Wilberg, A., Kok, H., & Lyus, R. 2009. Controlling crystallization and its absence : proteins, colloids and patchy models. *Physical Chemistry Chemical Physics*, **9**, 2197–2205.

Doye, J.P.K., & Wales, D. J. 1997. Structural consequences of the range of interatomic potential: A menagerie of clusters. *Journal of Chemical Society, Faraday Transactions*, **93**(24), 4233–4243.

E. Cuevas, F. Sencion-Echauri, D. Zaldivar, & Perez, M. 2013. Image Segmentation Using Artificial Bee Colony Optimization. *In:* Kacprzyk, J, & Jain, L (eds), *Handbook of Optimization : From Classical to Modern Approach*. Springer-Verlag Heidelberg: Springer.

Eyckelhof, C. J., & Snoek, M. 2002. Ant systems for a dynamic TSP: Ants caught in a traffic jam. *Pages 88–99 of:* Dorigo, M, Caro, G Di, & Sampels, M (eds),

Ant Algorithms : Third International Worshop, ANTS 2002, Lecture Notes in Computer Science, vol. 2463. Berlin: Springer.

Falk, J. E., & Soland, R. M. 1969. An algorithm for separable nonconvex programming problems. *Management Science*, **15**, 550–569.

Fantoni, R., Giacometti, A., Sciortino, F., & Pastore, G. 2014. Cluster Theory of Janus Particles. *Soft Matter*, **7**, 2419–2427.

Fejer, S., & Wales, D. J. 2015. Design of a Kagome Lattice from soft anisotropic particles. *Soft Matter*, 6663–6668.

Giacometti, A., Romano, F., & Sciortino, F. 2012. Theoretical Calculations of Phase Diagram and Self-Assembly of Patchy Colloids. *In:* Jiang, S., & Granick, S. (eds), *Janus Particle Synthesis, Self-Assembly and Applications*. Cambridge, UK: The Royal Society of Chemistry.

Girifalco, L., & Weizer, V. 1959. Application of the Morse Potential Function to cubic metals. *Physical Review*, **114**(3), 687–690.

Greenwood, G. 1999. Revisiting the Complexity of Finding Globally Minimum Energy Configurations in Atomic Clusters. *Zeitschrift fur Physikalische Chemie*, **211**, 105–114.

Gregurick, S., Alexander, M., & Hartke, B. 1996. Global Geometry Optimization of $(Ar)_n$ and $B(Ar)_n$ clusters using a modified Genetic Algorithm. *The Journal of Chemical Physics*, **104**(7), 2684–2691.

Grosso, A., Locatelli, M., & Schoen, F. 2007. A population-based approach for hard global optimization problems based on dissimilarity measures. *Mathematical Programming Series A*, **110**, 373–404.

Hartke, B. 1999. Global cluster geometry optimization by a phenotype algorithm with niches:Location of elusive minima and low-order scaling with cluster size. *Journal of Computational Chemistry*, **20**, 1752–1759.

Hassan, H., Khalid, M., & Imran, K. 2010. Intelligent Object and Pattern Recognition using Ensembles in Back Propagation Neural Network. *International Journal of Electrical and Computer Sciences*, **10**, 47–54.

Hendrawan, Y., & Murase, H. 2011. Neural-Intelligent Water Drops algorithm to select relevant textural features for developing precision irrigation system using machine vision. *Computers and Electronics in Agriculture*, **77**, 214–228.

Hestenes, M. R., & Stiefel, E. 1953. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, **49**, 409–436.

Hoang, D. C., Kumar, R., & Panda, S. K. 2012. Optimal data aggregation tree in wireless sensor networks based on intelligent water drops algorithm. *Wireless Sensor Systems IET*, **2**, 282–292.

Hoare, M. 1979. Structure and Dynamics of Simple Microclusters. *Pages 49–135 of:* Prigogine, I, & Rice, S (eds), *Advances in Chemical Physics*, vol. 15. John Wiley and Sons.

Hoare, M., & Pal, P. 1971. Physical cluster mechanics : Statics and energy surfacesfor monoatomic systems. *Advances in Physics*, **20**, 161–196.

Hodgson, R.J.W. 2002. Particle Swarm Optimization Applied to the Atomic Cluster Optimization Problem. *Pages 68–73 of: Proceedings of the Genetic and Evolutionary Computation.*

Holden, N., & Freitas, A. A. 2005. A hybrid Particle Swarm/Ant Colony Algorithm for the classification of hierarchical biological data. *Pages 100–107 of: Proceedings of the 2005 IEEE Swarm Intelligence Symposium.*

Holistory. *January Holidays A list of Observances Celebrations.* [Online; accessed 5-August-2015].

Holland, J. 1975. *Adaptation in Natural and Artificial Systems.* Cambridge, Massachusettes, USA: MIT Press.

Hong, L., & Cacciuto, A. 2012. Self-Assembly of Amphiphilic and Dipolar Janus Particles. *In:* Jiang, S., & Granick, S. (eds), *Janus Particle Synthesis, Self-Assembly and Applications.* Cambridge, UK: The Royal Society of Chemistry.

Hu, J., Zhou, S., Sun, Y., Fang, X., & Wu, L. 2012. Fabrication, properties and applications of Janus particles. *Chemical Society Review*, **41**, 4356–4378.

Hu, S., & Gao, X. 2010. Nanocomposites with spatially separated functionalities for combined imaging and magnetolytic therapy. *Journal of American Chemical Society*, **132**(21), 7234–7237.

Humphrey, W., Dalke, A., & Schulten, K. 1996. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, **14**, 33–38.

Ingber, L. 1989. Very Fast Simulated Re-annealing. *Mathematical and Computer Modelling*, **12**(8), 967–973.

Islam, M. R., & Rahman, M. S. 2013. An Improved Intelligent Water Drop Algorithm for a Real-Life Waste Collection Problem. *Pages 472–479 of:* Tan, Y, Shi, Y, & Mo, H (eds), *Proceedings of the 4th International Conference in Swarm Intelligence : Advances in Swarm Intelligence*. Springer.

Iwamatsu, M., & Okabe, Y. 2004. Basin-Hopping with Occasional Jumping. *Chemical Physics Letters*, **439**(4-6), 396–400.

Jatmiko, W., Ikemoto, Y., Matsuno, T., Fukuda, T., & Sekiyama, K. 2005. Distributed Odor Source Localization in Dynamic Environment. *IEEE Sensors*, 254–257.

Jiang, S., & Granick, S. 2012. Preface - An Introduction to Janus Particles. *In:* Jiang, S., & Granick, S. (eds), *Janus Particle Synthesis, Self-Assembly and Applications*. Cambridge, UK: The Royal Society of Chemistry.

Jin, N., & Rahmat-Samii, Y. 2007. Advances in Particle Swarm Optimization for Antenna Designs: Real Number, Binary, Single-Objective and Multiobjective Implementations. *Pages 556–567 of: IEEE Transactions on Antennas and Propagation*, vol. 55.

Johnston, R. 2003. Evolving better nanoparticles: Genetic algorithms for optimising cluster geometries. *Dalton Transactions*, 4193–4207.

Kan, A. H. G. Rinnooy, & Timmer, G. T. 1984. Stochastic Methods for Global Optimization. *American Journal of Mathematical and Management Sciences*, **4**, 7–40.

Kennedy, J., & Eberhart, R. 1995. Particle Swarm Optimization. *Pages 1942–1948 of: Proceedings IEEE Conference on Neural Networks*, vol. 4. IEEE.

Kern, N., & Frenkel, D. 2003. Fluid-fluid coexistence in colloidal systems with short-ranged strongly directional attraction. *Journal of Chemical Physics*, **118**, 9882–9889.

Kolossvary, I., & Bowers, K. J. 2010. Global optimization of additive potential energy functions: Predicting binary Lennard-Jones clusters. *Physical Reviews E*, **82**, 56711.

Korb, O., Stutzle, T., & Exner, T. E. 2007. An ant colony optimization approach to flexible protein-ligand docking. *Swarm Intelligence*, **1**, 115–134.

Lamport, L. *LaTex*.

Leary, R. 2000. Global Optimization on Funneling Landscapes. *Journal of Global Optimization*, **18**, 367–383.

Lee, J., Lee, I., & Lee, J. 2003. Unbiased Global Optimization of Lennard-Jones Clusters for $N \leq 201$ using the Conformational Space Annealing Method. *Physical Review Letters*, **91**, 080201.

Li, Z., & Scheraga, H. 1987. Monte Carlo-Minimization Approach to the Multiple-minima Problem in Protein Folding. *Proceedings of the National Academy of Sciences of the United States of America*, **84**(19), 6611–6615.

Li, Z., Lu, Z., Zhu, Y., Sun, Z., & An, L. 2012. A simulation model for soft triblock Janus particles and their ordered packing. *RSC Advances*, 813–822.

Liberti, L. 2006. *Introduction to Global Optimization.*

Liberti, L., & Kucherenko, S. 2005. Comparison of deterministic and stochastic approaches to global optimization. *International Transactions in Operational Research*, **12**, 263–285.

Liu, D., & Nocedal, J. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming B*, **45**, 503–528.

Locatelli, M., & Schoen, F. 2001. Fast Global Optimization of Difficult Lennard-Jones Clusters. *Computational Optimizations and Applications*, **21**(1), 55–70.

Locatelli, M., & Schoen, F. 2003. Efficient algorithms for large scale global optimization: Lennard-Jones clusters. *Computational Optimization and Applications*, **26**(2), 173–190.

Maier, R., Rosenx, J., & Xue, G. 1992. A discrete continuous algorithm for molecular energy minimization. *Pages 778–786 of: Proceedings of the IEEE Supercomputing.*

Manby, F. R., Johnston, R. L., & Roberts, C. 1998. Predatory Genetic Algorithms. *Communications in Mathematical and in Computer Chemistry*, **38**, 111–122.

Marques, J. M. C., & Pereira, F. B. 2010. An Evolutionary Algorithm for Global Minimum Search for Binary Atomic Clusters. *Chemical Physics Letters*, **485**(1), 211–216.

MATLAB. 2013. *version 8.1.0.604 (R2013a).* Natick, Massachusetts: The MathWorks Inc.

Meza, J. C., & Martinez, M. L. 1994. Direct search methods for the molecular conformation problem. *Journal of Computation Chemistry*, **15**, 627–632.

Miller, G. F., Todd, P. M., & Hegde, S. U. 1993. Designing neural networks using genetic algorithms. *In:* Schaffer, J D (ed), *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kauffman.

Montana, D. J., & Davis, L. D. 1989. Training feedforward networks using genetic algorithms. *In: Proceedings of the International Joint Conference on Artificial Intelligence*. Morgan Kauffman.

Nagalakshmi, P., Harish, Y., Kranthi, R., & Chakravarthi, J. 2011. Combined Economic and Economic Load Dispatch using Intelligent Water Drops-Continuous Optimization Algorithm. *Pages 168–173 of: Proceedings of the 2011 International Conference on Recent Advancements in Electrical, Electronics and Control Engineering*.

Neethling, M., & Engelbrecht, A. P. 2006. Determining RNA Secondary Structure using Set-Based Particle Swarm Optimization. *Pages 1670–1677 of: IEEE Congress on Evolutionary Computation*.

Nelder, J. A., & Mead, R. 1965. A simplex method for functional minimization. *Computer Journal*, **7**, 308–313.

Niesse, J., & Mayne, H. 1996. Global Geometry Optimization of Atomic Clusters Using a Modified Genetic Algorithm in Spaced-Fixed Coordinates. *Journal of Chemical Physics*, **105**, 4700–4706.

Nisisako, T., Torii, T., Takahashi, T., & Takizawa, Y. 2006. Synthesis of Monodisperse Bicolored Janus Particles with Electrical Anisotropy Using a Microfluidic Co-Flow System. *Advanced Materials*, **18**, 1152–1156.

Nocedal, J. *L-BFGS Fortran*. Available at `http://users.iems.northwestern.edu/~nocedal/lbfgs.html`.

Northby, J.A. 1987. Structures and binding of Lennard-Jones clusters : $13 \leq N \leq 147$. *Journal of Chemical Physics*, **87**(10), 6166–6177.

Okazaki, N. *libLBFGS: a library of Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS)*. [Online; accessed between January 2012 to April 2016].

Okazaki, N. 2012. *libLBFGS: a library of Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS)*. Available at `http://www.chokkan.org/software/liblbfgs/`.

Packard, N. H. 1990. A genetic learning algorithm for the analysis of complex data. *Complex Systems 4*, **5**, 543–572.

Pereira, F. B., Marques, J. M. C., Leitao, T., & Tavares, J. 2008. Designing Efficient Evolutionary Algorithms for Cluster Optimization: A study on Locality. *Pages 223–250 of:* Siarry, P., & Michalewicz, Z. (eds), *Advances in Metaheuristics for Hard Optimization.* Springer Berlin.

Petalas, Y. G., & Vrahatis, M. N. 2004. Parallel Tangent Methods with Variable Stepsize. *Pages 1063–1066 of: Proceedings of the 2004 International Joint Conference on Neural Networks*, vol. 2. IEEE.

Pullan, W. 2010. Unbiased Geometry Optimisation of Morse Atomic Clusters. *Pages 4496–4502 of: Proceedings of the WCCI 2010 IEEE World Congress on Computational Intelligence.* IEEE.

R Core Team. 2013. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

Raczynski, P., & Gburski, Z. 2005. The search for minimum potential energy structurs of small atomic clusters. Application of the Ant Colony Algorithm. *Materials-Science Poland*, **23**, 599–606.

Roberts, C., Johnston, R. L., & Wilson, N. T. 2000. A genetic algorithm for the structural optimization of Morse clusters. *Theoretical Chemistry Accounts*, **104**, 123–130.

Schulze-Kremer, S. 1994. Genetic Algorithms for Protein Tertiary Structure Prediction. *IEE Colloquium on Applications of Genetic Algorithms.*

Sciortino, F., Giacometti, A., & Pastore, G. 2010. A numerical study of one-patch colloidal particles: from square-well to Janus. *Physical Chemistry Chemical Physics*, **12**, 11869–11877.

Shah-Hosseini, H. 2007. Problem solving by Intelligent Water Drops. *Pages 3226–3231 of: Proceedings of the IEEE Congress on Evolutionary Computation.*

Shah-Hosseini, H. 2011. An approach to continuous optimization by the Intelligent Water Drops algorithm. *Pages 224–229 of:* Rahbar-Shamskar, Alireza (ed), *Proceedings of the 4th International Conference of Cognitive Science.* Elsevier.

Shymgelska, A., & Hoos, H. 2005. An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar proten folding problem. *BMC Bioinformatics*, **6**(30).

Sicher, M., Mohr, S., & Goedecker, S. 2011. Efficient moves for global optimization methods and their application to binary systems. *Journal of Chemical Physics*, **134**, 44106.

Singh, T. P. 2013. Face Recognition by using Feed Forward Back Propagation Neural Network. *International Journal of Innovative Research in Technology and Science*, **1**, 13–19.

Stallman, R., & Steele, G. L. Jr. 1985. *Emacs*.

Steer, K., Wirth, A., & Halgamuge, S. 2009. The Rationale Behind Seeking Inspiration from Nature. *In:* Chiong, R (ed), *Nature-Inspired Algorithms for Optimisation*. Springer-Verlag Berlin Heidelberg: Springer.

Stevenson, Paul. 2012. *RANDOM_SEED*. `http://web.ph.surrey.ac.uk/fortweb/glossary/random_seed.html`.

Stillinger, F. H., & Stillinger, D. K. 1990. Cluster Optimization simplified by interaction modification. *The Journal of Chemical Physics*, **93**, 6106–6107.

Synytska, A., Khanum, R., Ionov, L., Cherif, C., & Bellman, C. 2011. Water Repellent Textile via Decorating Fibers with Amphiphilic Janus Particles. *Applied Materials and Interfaces*, **3**(4), 1216–1220.

Tao, Y., Ruchu, X., & Wenqi, H. 2011. Global Opt of Binary Lennard-Jones Clusters Using Three Perturbation Operators. *Journal of Chemical Information and Modeling*, **53**(1), 572–577.

Tomson, P., & Greenwood, G. 2005. Using Ant Colony Optimization to Find Lower Energy Atomic Cluster Structures. *Pages 2677–2682 of: Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, vol. 3. IEEE.

Wales, D. J. *GMIN*. Available at `http://www-wales.ch.cam.ac.uk/software.html`.

Wales, D. J. 2003. *Energy Landscapes*. Cambridge: Cambridge University Press.

Wales, D. J., & Doye, J. 1997. Global optimization by Basin Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms. *Journal of Physical Chemistry A*, **11**(28), 5111–5116.

Wales, D. J., Doye, J. P. K., Dullweber, A., Hodges, M.P., Naumkin, F. Y., Calvo, F., Hernandez-Rojas, J., & Middleton, T. F. *The Cambridge Cluster Database*.

Weise, T. 2009. *Global Optimization Algorithms - Theory and Applications*. Self-published.

Wille, L.T., & Vennik, J. 1985. Computation complexity of the ground-state determination of atomic clusters. *Journal of Physics A: Mathematical and General*, **18**(8), 419–422.

Williams, T., & Kelley, C. 2010 (March). *Gnuplot 4.4: an interactive plotting program.*

Wolf, M.D., & Landman, U. 1998. Genetic Algorithms for Structural Cluster Optimization. *Journal of Physical Chemistry A*, **102**, 6129–6137.

Wu, L., Ross, B., Hong, S., & Lee, L. 2010. Bioinspired Nanocorals with Decoupled Cellular Targeting and Sensing Functionality. *Small*, **6**(4), 503–507.

Xiang, Y., Jiang, H., Cai, W., & Shao, X. 2004. An Efficient Method Based on Lattice Construction and the Genetic Algorithm for Optimization of Large Lennard-Jones Clusters. *Journal of Physical Chemistry A*, **108**, 3586–3592.

Xing, B., & Gao, W. 2013. *Innovative Computational Intelligence : A Rough Guide to 134 Clever Algorithms.* Springer Cham Heidelberg New York Dordrecht London: Springer.

Xue, G. 1994a. Improvement on the Northby Algorithm for Molecular Conformation : Better Solutions. *Journal of Global Optimization*, **4**, 425–440.

Xue, G. 1994b. Molecular Conformation on the CM-5 by Parallel Two-Level Simulated Annealing. *Journal of Global Optimization*, **4**, 187–208.

Zhao, Y., Gu, H., Xie, Z., Shum, H., Wang, B., & Gu, Z. 2013. Bioinspired Multifunctional Janus Particles for Droplet Manipulation. *Journal of American Chemical Society*, **135**, 54–57.

Zhou, T., Bai, W., Cheng, L., & Wang, B. 2005. Continuous extremal Optimization for Lennard Jones clusters. *Physical Review E*, **72**, 16702.